JPL Publication 90-16

Operations Mission Planner

Final Report

Eric Biefeld
Lynne Cooper

March 15, 1990

Prepared for

U.S. Department of Defense

Through an agreement with

National Aeronautics and
Space Administration

by

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

JPL Publication 90-16

Operations Mission Planner

Final Report

Eric Biefeld
Lynne Cooper

March 15, 1990

Prepared for

U.S. Department of Defense

Through an agreement with

National Aeronautics and
Space Administration

by

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

The research was performed by the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by the United States Department of Defense through an agreement with the National Aeronautics and Space Administration.

CONTENTS

ABSTRACT

The Operations Mission Planner (OMP) Final Report documents the findings of the OMP research task, which investigated the applicability of Artificial Intelligence (AI) technology in support of automated scheduling. This report summarizes the goals of the effort and highlights the technical accomplishments. The OMP task succeeded in identifying how AI technology could be applied and demonstrated an AI-based automated scheduling approach through the OMP prototypes.

EXECUTIVE SUMMARY

The Operations Mission Planner (OMP) task was a research effort to determine if and how Artificial Intelligence (AI) technology could be used to support automated mission scheduling. The domain was a resource-allocation problem that was highly oversubscribed, which required real-time reaction to new tasking and changes in the operational environment, and the solution of which required minimal perturbation of the existing schedule as a result of those reactions. The goal for the automated scheduler was to minimize the number of tasks not accomplished by the schedule.

The question asked by the OMP task sponsor was if and how AI technology could be used to support automated scheduling within the constraints imposed by the problem domain. There are two components to this question: those of general interest in automated scheduling and those that are domain specific. To fully answer the sponsor's question, the OMP research had to identify the problem areas and then employ the OMP system to resolve them. The results indicate that AI technology can be effectively applied to address the automated scheduling problem, but only with the introduction of new techniques and methodologies.

The approach to automated scheduling developed on OMP is based on the process used by expert human schedulers and employs several new AI-based scheduling techniques. The major innovation is the incorporation of true multipass scheduling or iterative refinement, whereby the automated scheduler builds and refines a schedule over a series of passes, learning from each pass and modifying its approach as it learns more about the schedule being developed. In this state-of-the-art approach, the OMP system allows the schedule to contain conflicts and to modify its scheduling actions based on the identification and classification of these conflicts. The use of iterative refinement, in turn, has necessitated the development of new types of domain representation, control mechanisms, and chronology development.

The OMP approach was validated by the development of two distinct prototypes, OMP I and OMP II. Developed during the first year of the project, OMP I provided a means of evaluating the iterative refinement approach. OMP II, a unique implementation, added the control mechanisms necessary to interleave the phases of the iterative refinement approach and demonstrated the various supporting AI technologies. The OMP II prototype was demonstrated using a scenario scaled to closely approximate the demands of an operational environment. The demonstration showed that OMP II could produce a valid schedule in real time (minutes) and could adjust the schedule when new tasks were added, when old tasks were changed, or when changes, also in real-time (seconds), occurred in the available resources.

The adoption of the OMP system has prompted new questions and added problems that should be addressed through ongoing, longer term scheduling research efforts. However, even without addressing those problems, the OMP system can be used to support operational domains. The research successfully demonstrated the applicability of state-of-the-art AI technology and indicated future research needed to further advance the concepts presented.

1.0 INTRODUCTION

The Operations Mission Planner (OMP) task is a research effort to explore the potential application of Artificial Intelligence (AI) technology to automated planning and scheduling. The OMP task terminated in September, 1989, at the convenience of the sponsor.

This report covers the progress made on the OMP task throughout its two-year effort and contains an overview of the task and its objectives, an account of the progress made toward those objectives, and a summary of the technical accomplishments. The report provides a summary of the year-one goals and accomplishments but concentrates primarily on year two. The primary purpose of this report is to describe the technical accomplishments of the OMP task with respect to the sponsor's goals and highlight the capabilities of the OMP II prototype automated scheduling system. The main section summarizes the technical issues; additional details on the AI technologies used in the OMP approach are presented in the appendixes.

The OMP research demonstrated the significant potential for the application of AI technology in automated scheduling. The innovative approach, Iterative Refinement, based on the techniques used by expert human schedulers, was combined with advanced concepts from automated scheduling research in the manufacturing job shop domain [see Appendix I] . The resulting OMP prototype provided a significantly different approach to automated scheduling, one which has demonstrated the potential to meet the needs of the OMP sponsors for nonnervous, real-time schedule generation and event handling. The incorporation of knowledge into the search process allows the scheduler to reduce the search space and thereby increase the efficiency of the scheduling process. In addition, the advanced domain-representation techniques employed in OMP hold significant promise for representing real-world domains.

2.0 PROBLEM DESCRIPTION

The question asked by the OMP task sponsor was if and how Artificial Intelligence Technology could be used to support automated scheduling within the constraints imposed by the sponsor's problem domain. There are two components to this question: those of general interest in automated scheduling and those that are domain specific. To fully answer the sponsor's question, pertinent issues in each area had to be identified and resolved. The OMP research has identified the problem areas, the OMP system was designed to mitigate them, and the results demonstrate that AI technology can effectively be applied to address the automated scheduling problem, but only with the introduction of new techniques and methodologies.

2.1 General Problem

The objective for automated scheduling is to develop and maintain schedules within which a set of tasks can be accomplished while satisfying a set of temporal and resource constraints. Unfortunately, given the complexity of real-world domains, existing automated planning and scheduling systems are unable to satisfy these requirements.

Early research into automated planning systems evolved from general-purpose problem-solving techniques. These techniques depend largely on search mechanisms that explore vast solution spaces. Since the scheduling search problem is inherently intractable, the computation times are long, with no guarantee of finding an acceptable, much less an optimal, solution.

Additional problems include the inability to represent complex domains, the inability to react to changes in the planning environment, the inability to produce schedules for large numbers of tasks, and the inability to produce efficient[1] schedules in oversubscribed domains.

Advanced research into the use of artificial intelligence in the planning/scheduling process is aimed at reducing search times while creating more efficient schedules. This research has centered on the use of heuristics to analyze the problems in a developing schedule and the efforts necessary to resolve them (i.e., to guide the search process). Additional information about classical and advanced planning research is available in [1].

OMP research analyzes the difficulties associated with automated planning and scheduling. The basic premise of this research is that both general-purpose and domain-specific knowledge can effectively be used to contain a search and increase the effectiveness of the scheduling process.

2.2 Problem Domain

The problem domain for the OMP research task was the Foreign Broadcast Information Service (FBIS). FBIS is responsible for monitoring and collecting foreign broadcasts throughout the world and returning intelligence information. The FBIS scheduling problem devolves into one of allocating a finite set of resources to collect the maximum amount of information. Because there is an order of magnitude more information requested than FBIS could possibly support, the problem domain is highly oversubscribed. The collection process is influenced by the enforcement of a strict priority system and the requirement to support real-time events affecting the schedule during execution.

The functional requirements for an automated scheduling system for the FBIS domain are to

(1) Minimize lost collection.
(2) Minimize perturbation of collection.
(3) Perform continuous forward evaluation.
(4) Plan with knowledge from previous decisions.
(5) React quickly.

For a complete description of the FBIS scenario, refer to Appendix B.

3.0 OVERVIEW OF ACCOMPLISHMENTS

The OMP task demonstrated the significant potential for the application of AI technology to automated planning and scheduling. The OMP prototype featured many special capabilities that were enabled by the use of AI. The major innovation was the incorporation of the iterative refinement scheduling concept.

[1] In an oversubscribed domain, it is impossible to accomplish all tasks. Therefore, the efficiency of the schedule is relative; it is judged by how many tasks the schedule can accomplish.

The other major technical accomplishments of the OMP task were strongly influenced by the iterative refinement concept. Assessment heuristics determine the state of the schedule. Executive control mechanisms make use of these assessments to identify the appropriate strategies to employ during a given phase and to determine when to change phases. The control strategies identify which problems to resolve and set the context to determine what low-level scheduling actions to perform. The results of these scheduling actions then feed into a chronology system that provides the information necessary for the assessment heuristics to perform their functions. A detailed domain representation supports the variety of tactics necessary to perform advanced scheduling. The entire process depends on a Knowledge-Intensive Search to avoid intractability problems. Advanced user-interface techniques allow visibility into the system and enable the creation of additional heuristics, strategies, and tactics.

The accomplishments of OMP in each of these areas are presented in the following sections.

3.1 Iterative Refinement

Iterative refinement is the cornerstone of the OMP approach. Based on the techniques used by the expert human schedulers in the interplanetary exploration domain [see Appendix G], OMP makes a series of passes over the schedule. This technique deviates substantially from the classical approach to automated planning typified by research systems such as FORBIN [2], Deviser [3], NONLIN [4], and ISIS [5]. These classical schedulers function by incrementally building a schedule; new tasks can be added to the schedule if they do not result in any conflicts. Therefore, when a conflict caused by a previous scheduling action is discovered after several additional scheduling actions, the scheduler undoes the schedule up to that point (backtracks) and discards any of the newer scheduling actions. This philosophy rapidly degenerates into a depth-first search through a vast solution space, an inherently intractable problem. Illegal schedules (those containing conflicts) were never allowed.

OMP uses a multiphase approach that enables a schedule to be developed and modified (improved) through a series of passes over the schedule. With the additional flexibility provided by interleaving these planning phases, the scheduler can choose to focus globally across the entire schedule or locally on a given area of the schedule, at its discretion.

The key innovations enabled by the iterative refinement approach are:

(1) Representing illegal schedules.
(2) Refining the schedule and resolving conflicts by modifying an existing schedule, not by building a new schedule from scratch.
(3) Using information about the state of the schedule from prior passes to guide the scheduling actions in subsequent passes.
(4) Using different strategies during different passes.
(5) Eliminating backtracking. If a poor decision is made, it can be unmade through the application of an appropriate action. The valid parts of the schedule developed after the mistake are not affected, although they may be causally related.

(6) Reducing replanning to selecting the appropriate scheduling phase and reinitiating planning with the additional tasks or resource changes.

The OMP approach follows the philosophy of first scheduling to identify the problems and then refining the schedule. In a significant departure from classical planners, OMP allows illegal schedules to develop and conflicts to exist within the schedule during its evolution. By first building schedules which help to identify the areas of high resource contention and task interaction, referred to as *bottlenecks*, OMP is able to focus its scheduling efforts. This parallels the expert human approach. Human schedulers first assign the tasks to the *timelines*, data structures representing resource usage and task assignments over time (Figure 1), to identify the potential problem areas. Once they have identified these areas, they begin focusing their efforts there, jumping back and forth from one area to another to narrow down the problems. A conflict-free schedule is produced only as a final output.

A major benefit of the iterative refinement approach is the simplification of the replanning problem. Any changes in tasking or environment can be dealt with by the event-handling phase, which assesses the potential impact of the changes on the schedule and reinitiates planning in one of the previous phases. All the knowledge about the schedule up to that point is retained, and areas not directly affected by the changes are modified only if absolutely necessary. This capability supports nonnervous rescheduling, an important requirement when a schedule is not just an end product, but one which also serves as an input to other processes.

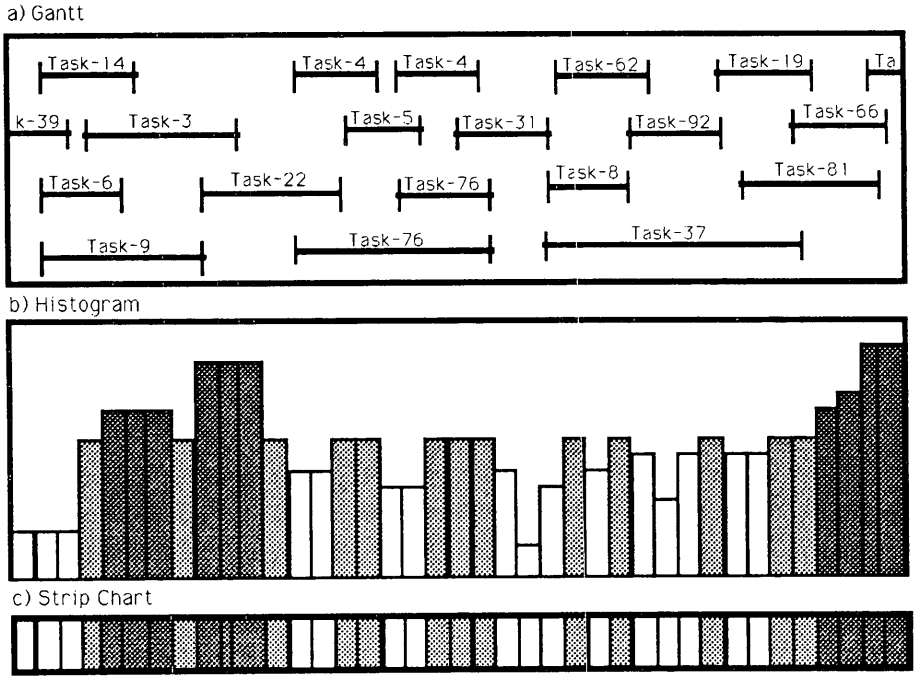


Figure 1. Timelines

3.2 Assessment Heuristics

OMP uses information on the types of scheduling actions performed and the results of these actions to assess the state of the schedule. The assessment heuristics identify and classify the bottlenecks of the schedule. Control mechanisms then use these assessments to change to a new scheduling phase or modify the strategies and tactics used during the current phase. The assessment and classification heuristics that OMP has implemented are for bottleneck identification. As discussed in OPT [6] and OPIS [7], bottlenecks represent the most difficult areas of the schedule to complete. By identifying these bottlenecks and taking appropriate measures to resolve problems in them, OMP significantly narrows its search space and reduces the overall effort required to produce a schedule.

3.3 Multilevel Control

OMP has a wide range of flexibility because of its multilevel control. It is not tied to a single, all-encompassing, general-purpose control algorithm. OMP's three levels of control are Executive (or Master), Strategic, and Tactical (Figure 2). The Executive Level is responsible for initiating scheduling, determining the appropriate scheduling phase, and determining when to terminate scheduling. The Executive sets the global context for any scheduling actions and activates the appropriate strategies.

The Strategic Level determines which of the possible tactics are appropriate, establishes the parameters under which the tactics operate, and determines where to focus the search. The Tactical Level, in turn, identifies possible low-level scheduling actions and uses heuristics to fill in the parameters of the possible scheduling actions.

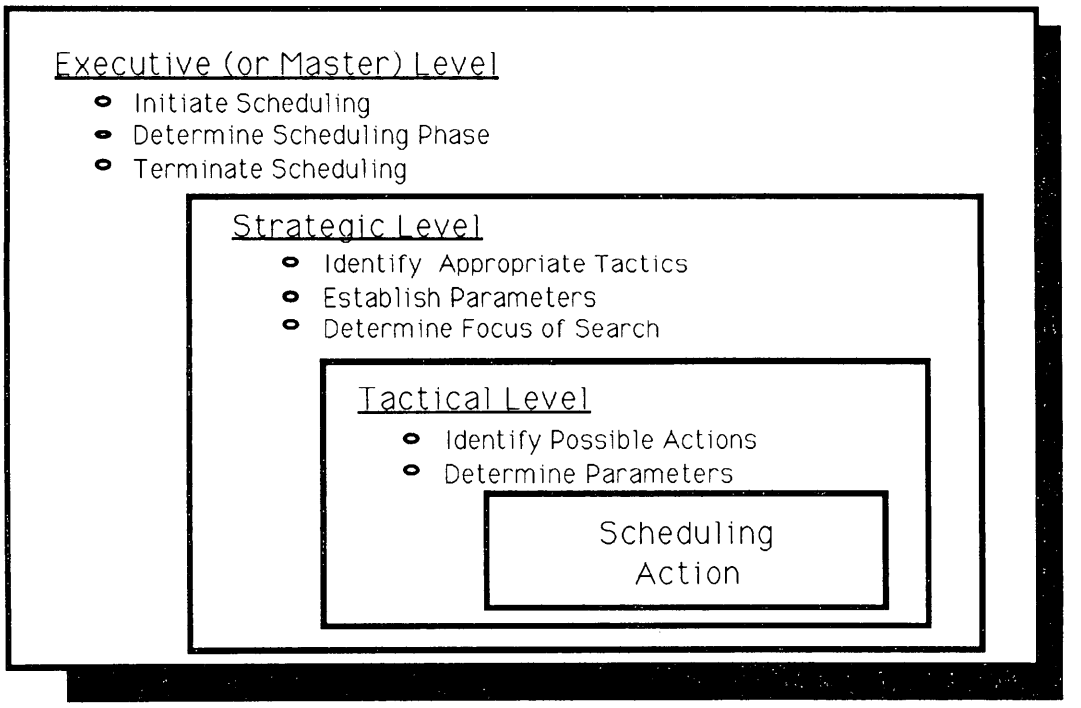


Figure 2. OMP Multilevel Control

Multilevel control enables OMP to defer the details of some scheduling decisions to lower level mechanisms. This has the effect of focusing the search on a reduced area and automatically choosing from only those techniques that are applicable. The flexibility that OMP gains thereby allows OMP to expend a greater amount of effort in critical areas without conducting large global searches.

3.4 Chronology

The "feedback" mechanism that OMP uses to determine bottlenecks is the chronology system [see Appendix H]. *Chronologies* are limited histories of the scheduling activities that have been performed and their effects on the schedule. Chronologies provide the information required to support the assessment heuristics and are currently tied directly to the resources. The parameters that are tracked relate to how scheduling actions affect conflict levels. Currently, chronologies are used in bottleneck identification and in supporting executive-control functions.

The OMP concept has also included chronologies tied to the individual tasks. Due to programmatic time constraints, these types of chronologies were not explicitly implemented. However, a side effect of the extended task representation discussed in Section 3.5, and the integration of lower level control structures, is that some chronology-type features are implicitly invoked. For example, heuristics keep track of certain types of scheduling actions, such as right shifts, and ensure that inverse actions, e.g., left shifts, are not considered as a next step. This enables OMP to avoid circular searches.

3.5 Domain Representation

OMP's task representation is extremely rich and is the basis for OMP's flexibility. Tasks are represented using a detailed activity structure that hierarchically depicts the components of a task. At each level of the activity tree (Figure 3), different decisions about the allocation of resources are made. Higher level nodes correspond to the more global characteristics of the given task and, consequently, those which would have the most impact on other components of the schedule. Lower level nodes focus on the breakdown and modification of the task itself, enabling the use of advanced scheduling actions. The leaves of the activity tree specify the actual resource assignments.

The activity trees interact with the control structure by providing a framework in which control decisions can be made. Any constraints on the task itself are manifested in its activity tree. Therefore, control structures are prevented from performing any scheduling actions that would cause an internal inconsistency.

Each node of the activity tree has an associated set of tactics. That set is narrowed down according to the scheduling context. When a strategy decides to reschedule a particular task, it sets the context. The appropriate tactics then suggest actions to modify the activity structure by reassigning resources or modifying the task parameters. The search engine then chooses from the set of suggested actions.

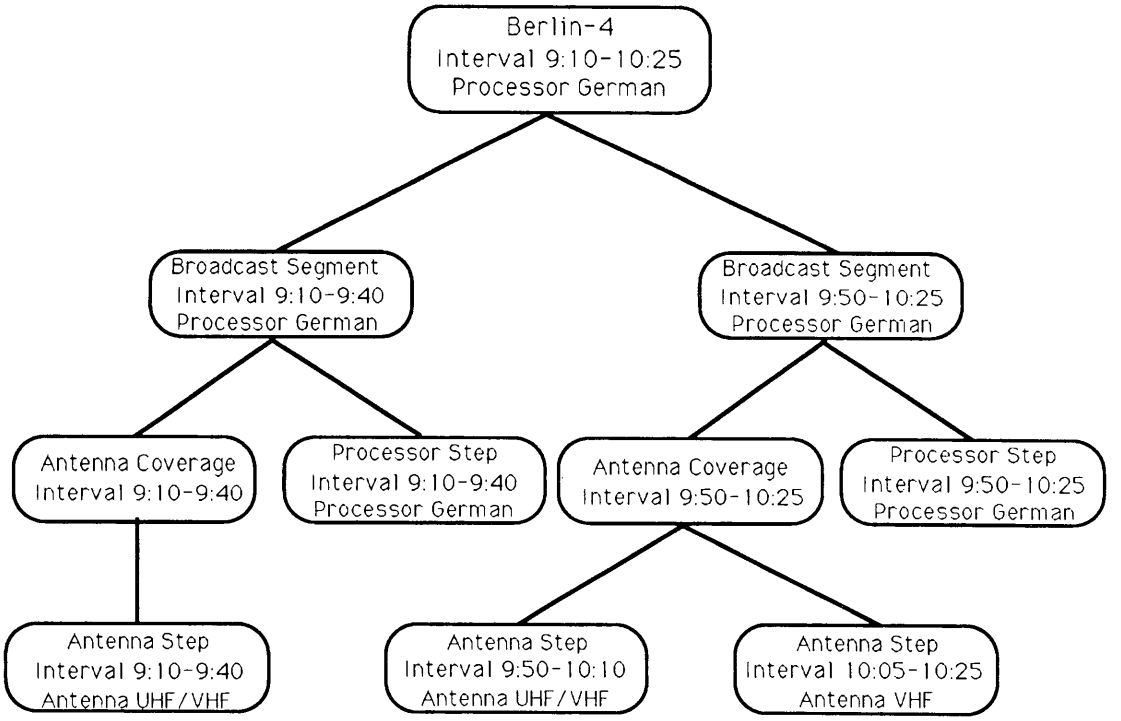


Figure 3. Activity-Tree Structure

3.6 Knowledge-Intensive Search

The enhanced task representation described in Section 3.5 required the development of a search engine that could search through the different representations of the task and make use of the advanced techniques available. The search engine is highly dependent upon a strategy that sets up the proper context for the search and on tactics that respond correctly. If this does not occur, the problem rapidly devolves into a depth-first search over a large solution space, an intractable problem. To prevent this from happening, the search engine makes use of the knowledge available on the schedule and the tasks and performs a Knowledge-Intensive Search.

An example of the type of knowledge that the search engine uses is the location of bottleneck areas. To resolve these areas, an intensive search is necessary. If uncontrolled, however, the search becomes intractable. The search engine limits the paths that the search can take and the depth along any given path using (1) its knowledge concerning the boundaries of the bottleneck region, (2) the types of actions it can perform, (3) any constraints imposed by the strategic control mechanism, and (4) the constraints internal to tasks. These tightly controlled depth and breadth cutoffs prevent the search from becoming intractable.

3.7 Advanced User Interface

The development of an advanced user interface was an integral part of the OMP task. It was necessary to enable the developers to assess OMP's progress toward completing a schedule. The interface consists of several interactive graphical displays that make extensive use of color coding as a means of providing additional information in a compact form.

The OMP interface was important in the development and debugging of the heuristics. Progressing beyond simple tabular displays to timelines, directional timelines, histograms, and strip charts enabled the developers to see, in a much more immediate fashion, what OMP was doing. Given the large number of tasks and severe oversubscription of the system in the test scenarios, the graphic display proved to be a labor- and time-saving device.

3.8 Summary

OMP's advances are due to its adoption of an iterative refinement approach. This approach required the development of highly interdependent advanced concepts in control and task representation. Chronologies are needed to support assessment heuristics that are required to support the multilevel control mechanisms, that in turn are necessary to support a knowledge-intensive search that is necessary due to the enhanced task representation. The integration of all these pieces into a working model of an iterative refinement scheduling system is the most significant achievement of the OMP task.

The AI technology accomplishments of the OMP task were necessary to support the specific requirements of the FBIS domain. Table 1 shows the relationship between the requirements and technology associated with each requirement. In addition, Table 1 identifies the driving characteristics of the the technologies they address.

Requirement	Drivers	Technology
1) Manage Lost Collection	• Operate in Over-Subscribed Domain • Control Depth of Search • Reevaluate Previously Deleted Items • Pack Tasks into Schedule	• Representation Technologies Allowing Conflicts in Schedule • Focused Knowledge-Intensive Search • Optimize Phase • Advanced Gapping, Shrinking, Hand-off techniques

2) Minimize Perturbation of Collection	• Prevent "Planning from Scratch"
	• Address Alerts
	  • Yes/No Decision
	  • Minimize Lost Collection
		• Interactive Refinement
		• Event Handler
		  •• Slash and Burn
		  •• Optimize Phase

3) Perform Continuous Forward Evaluation	• Assess State of Schedule	• Chronology • Assessment Heuristics
4) Plan with Knowledge from Previous Decisions	• Use Knowledge to Guide Search	• Knowledge-Intensive Search • Strategies and Tactics • Assessment Heuristics • Actively Structure for Constraint Representation
5) React Quickly	• Real-Time (seconds) Response	• Interactive Refinement • Event Handler

Table 1. Requirements, Drivers, and Technology

4.0 STATUS

The primary objective of the OMP effort was to design a system capable of meeting the performance goals identified in Section 2.1 and to identify AI techniques that could be applied successfully in the problem domain. The viability of the approach taken in the OMP design would be proven through the development of a successive series of prototype-scheduling systems that would incorporate the appropriate AI technologies.

4.1 Objectives: Year One

The objectives for OMP during its first year consisted of formalizing the OMP scheduling problem; identifying requirements; and designing a prototype, OMP I, which would incorporate AI technology. The specific goals and accomplishments for year one are detailed in [8].

4.1.1 OMP I Prototype

Development of the OMP I prototype focused on providing an accurate and detailed representation of the FBIS problem domain and developing the basic scheduling algorithms that would enable OMP to perform iterative refinement. OMP I provided the ability to represent both capacity and direction resources (processors/relays and antennas, respectively). Tasks were represented as simple steps that would be assigned to the required resources. A simple loading mechanism was used to initialize the resource data structures, referred to as timelines. A random-shuffling algorithm was used to develop information on the particular scheduling problem. A preliminary form of chronology was developed. The chronologies were analyzed to identify bottlenecks in the scheduling process.

OMP I served as a proof of concept for the iterative refinement approach adopted in the OMP task. In this approach, the scheduler progresses through a series of different scheduling phases, making use of information gained during previous schedule development to guide the process toward completion. Each scheduling phase has specific objectives associated with it and uses a set of heuristics to accomplish those objectives.

OMP I had an advanced graphical user interface that incorporated multiple displays, color coding, and interactive features to support both development and use of the system. All the data structures for both resources and tasks were easily available to the user in a variety of formats.

OMP I was a major first step in the development of an AI-based scheduling approach to support FBIS-type domains. It served as a proof of concept for the iterative refinement approach and incorporated several AI techniques in both the representation and use of scheduling knowledge. The effectiveness of OMP I was demonstrated using the FBIS scenario described in Appendix B. Its limitations, particularly in the areas of detailed task representation, strategic and tactical control mechanisms, and bottleneck classification, formed the basis for the second year's research efforts. OMP I prototyped the initial load and resource-centered phases and part of the time-centered phase, and used simplistic control structures.

4.2 Objectives: Year Two

The primary objective for year two was to develop an enhanced prototype, OMP II, which was based on OMP I and incorporated advanced AI techniques. The specific objectives were to

(1) Augment the OMP I implementation of tasks, resource timelines, and interface features.
(2) Research the use of chronologies for eras, tasks, and the schedule as a whole.
(3) Design and develop a scheduling engine that makes use of knowledge available about the schedule and that is focused by both general-purpose and domain-specific heuristics.
(4) Provide an executive-level control function.
(5) Provide an event-handling capability.
(6) Analyze the applicability of Operations Research techniques in AI-based planning and scheduling.
(7) Conduct quarterly reviews, submit quarterly reports, and conduct an annual demonstration.

4.2.1 Status of Objectives

The second year of the OMP task suffered from a delayed start and an accelerated end (due to uneven funding), which seriously affected the accomplishment of all the objectives originally set for the year. Objectives 1 through 5 were tailored to fit within the tighter constraints; Objective 6 was deleted; and Objective 7 was reduced to two quarterly reports and reviews, and a final demonstration. In addition, two objectives, namely prototyping the search engine and delivering the final report and demonstration, were moved from year three to year two. Some capabilities from OMP I were not implemented in OMP II, and the FBIS scenario was modified so that the major research topics could be addressed by the end of year two to coincide with the revised plan. Despite these changes, several technical breakthroughs, as described in Section 3, were achieved.

The following paragraphs specifically address the status of the year-two objectives. The capabilities implemented in the OMP II prototype are addressed in Section 4.2.2.

OMP II Design

The OMP II design augmented the OMP I design in several ways. Extensive effort went into enhancing the task representations. The resource timelines were updated to interact with these new representations, and new interface features including strip representation of histograms and message-and-editing windows were added. The problem domain was modified to include temporal flexibility, such as the addition of multiple windows of opportunity for the tasks. The control structures were enhanced to provide executive-level control and optimization.

OMP II Implementation

OMP II was virtually a complete reimplementation of OMP I, with minimal transfer of code. Due to the redesign of the task representations, the previously implemented scheduling

capabilities had to be updated. The OMP II implementation supports scheduling through all the iterative refinement phases and includes postprocessing support for determining statistics and developing a sequence of events based on the schedule.

Some representation capabilities from OMP I were not included in OMP II in order to continue the emphasis on the research goals of the task. Specifically, OMP II represents only capacity resources and cannot completely model the directional (state) resources (e.g., antennas).

Chronology System

Research into the application of chronology systems was limited to time segments on resource timelines, referred to as eras. The algorithm used in OMP II for developing chronologies trades accuracy for computational efficiency. Era chronologies are developed using simple algorithms established explicitly for tracking this information. Other, implicit chronology-type information relating to individual tasks is developed as a by-product of the new activity-structure paradigm used in OMP II for task representation. This information is encoded as context additions made by tactics during actual scheduling. Both the explicit and implicit chronology information are used by the control heuristics to direct the search process.

Scheduling Engine

Extensive work was done in year two to design and develop the OMP scheduling engine. The engine is based on the concept of a Knowledge-Intensive Search. The engine makes use of information gathered by the control strategies, control tactics, and the assessment heuristics to control the search process.

Executive-Level Control

Executive-level control, also referred to as master-level control, provides basic control over the five phases of the scheduling system. During schedule generation, the executive is responsible for determining the appropriate scheduling phase and selecting the strategies. The basic capability for event handling, which includes the introduction of new tasking (e.g., alerts) during schedule execution and changes in resource status (e.g., loss of a given antenna), was implemented during year two. This implementation causes OMP to be reinvoked into the Optimization Phase for alerts and into the Resource-Centered Phase for resource events.

4.2.2 OMP II Prototype

OMP II prototyping efforts focused on the areas of task representation, strategic and tactical control, and classification. OMP II has (1) a detailed activity structure, (2) top-level strategic control, (3) tactically controlled depth search, and (4) bottleneck identification and assessment heuristics. OMP II advances the iterative refinement approach by enabling interleaving of the different scheduling phases. The significance of each of these OMP II features is summarized in the following paragraphs and discussed in detail in Appendix A, the OMP Technical Report. The OMP II prototype implemented advanced versions of the resource-centered and bottleneck-centered phases. The necessary automated control structures were also implemented.

Task Representation

In OMP I, a very simple task representation was used. Tasks consisted of simple steps that could either be scheduled (allocated resources) or not. This representation prevented OMP I from using advanced scheduling strategies. The task representation developed for OMP II is much richer and more robust, and supports the advanced features that OMP I lacked, e.g., gapping, deleting part of the middle of a task, but catching the beginning and end; shrinking, deleting part of the beginning or end of a task, but catching the middle; and hand-offs, using one resource to catch the beginning of a task and another resource(s) to catch the remaining part.

The activity-tree task representation provides OMP II with a flexibility missing in OMP I. Scheduling, from the task perspective, devolves to modifying the activity tree. Constraint propagation, such as for temporal and resource dependencies, is automatically addressed as part of updating the task representation. Inconsistent activity structures are not allowed.

Strategic and Tactical Control

A second major advance in OMP II is the integration of strategic and tactical control mechanisms. Strategic control mechanisms are those that lay out the global constraints under which the tactics will perform scheduling actions. For example, strategic control mechanisms set the cutoff levels for oversubscription of the resources, determine whether global or local actions are permitted, and restrict the search depth. The strategies are responsible for assessing the state of the schedule and determining when to change scheduling phases.

Tactical control mechanisms are associated with each specific node of the activity tree. These tactics are the rules that guide the search engine. Based on the type of resource interaction, the context (e.g., global vs local modifications allowed) and the allowable scheduling actions (e.g., moves, deletes, gapping, etc.) suggest actions to resolve the problem. The tactics limit the depth of a search and add to the search context established by the strategies, thus preventing circular searches.

Strategic and tactical control mechanisms have been integrated into the OMP II search engine. Because of the limitations in both task representation and automated control mechanisms, the search in OMP I was very simple. Due to the increased complexity of OMP II, a search engine was built to take advantage of the advanced control mechanisms. The search engine depends on a strategy that sets up the proper context and on the tactics that perform appropriately within that context.

Classification of Bottlenecks

Both OMP I and OMP II use chronologies to identify bottlenecks. However, OMP II improves upon OMP I by also classifying the bottlenecks. The assessment heuristics assess the size, level of oversubscription, and loading characteristics of a given bottleneck to determine the appropriate strategies for resolving it.

OMP II currently has three levels of classification: (1) a large level of oversubscription, (2) temporally long with a slight amount of oversubscription, and (3) temporally short. Strategies are selected to resolve the bottlenecks based on these classifications. For a temporally

short bottleneck, an appropriate strategy is one that uses local modifications such as gapping, shrinking, and hand-offs first, and then resorts to deleting, if necessary. Large levels of oversubscription in a bottleneck indicate a need for a deletion strategy, while a temporally long and slightly oversubscribed bottleneck can be broken into smaller pieces and solved using local strategies.

The classification process is extremely important to the selection of strategies and therefore has a significant impact on the effectiveness of the scheduling engine. The three classifications identified in OMP II are gross categories that provide high-level guidance. Upper level control strategies, in turn, set the parameters by which the bottlenecks are identified and classified. Therefore, there is strong interaction between the bottleneck classification system and the higher level strategies that influence the effectiveness of the scheduling engine.

Interleaved Iterative Refinement

The OMP task introduced the concept of multiphase scheduling. During year one, scheduling was intended to progress serially through the five phases (Initial Load, Resource-Centered, Bottleneck-Centered, Optimization, and Event-Handling). This concept was upgraded during year two to allow interleaving of the phases. The sequence in which the phases are invoked (and reinvoked) depends on the current state of the schedule, the tasks remaining to be scheduled, and the focus state of the scheduler. While OMP I implemented only the Initial Load and Resource-Centered Phases, and simulated the other phases, OMP II has implemented (to varying degrees) all the planning phases and incorporates control mechanisms that enable interleaving.

The most obvious advantage of interleaving is that, unlike OMP I, OMP II can load the schedule until it reaches a problem threshold; it then resolves the problem to an acceptable level and continues loading. Since loading strategies are influenced by the priorities of the tasks, this ensures that all tasks, even the lowest priorities, will be given a fair chance to enter the schedule. Conflicts due to interactions between higher priority tasks will be resolved without arbitrarily affecting the lower priority tasks.

4.3 Objectives: Year Three

The following objectives were originally scheduled for the third year of work under the Operations Mission Planner contract but were deleted when the period of performance was reduced to two years:

(1) Design and develop the Optimizer Phase to include a depth-search engine and heuristic pruning.
(2) Develop a planning shell that would provide the necessary language to specify resources, tasks, and heuristics.
(3) Develop a prototype operations interface.
(4) Perform studies on the issues of centralized/decentralized scheduling, timing, manual override of automated planning, and complexity.
(5) Conduct quarterly reviews; publish quarterly reports, a final report, and conduct a final demonstration.

The depth-search engine (item 1) and the final report and demonstration (item 5) were moved to year two.

5.0 RECOMMENDATIONS

The OMP task has made significant progress in the area of automated scheduling research. The results have also raised new questions that could form the basis of advanced research. In addition, several areas originally planned as part of this effort were not addressed, due to the loss of funding. The following sections provide recommendations for follow-on activity to address research issues and applications issues.

5.1 Applications Issues

The following issues are related to the actual application of OMP to an operational domain. Although the OMP technology has matured to the point where it is ready for technology transfer, there are several applications-related issues that must be addressed.

5.1.1 Integration of State Resources

The OMP I Prototype used a special representation for the antenna resources. In the FBIS scenario, this representation addressed constraints that required an antenna to be pointed in a particular direction to support a task. Tracking of additional constraints (such as the amount of time required to pan an antenna to the given direction and the number of times the antenna cable could be wrapped around the pedestal before it had to be unwrapped) was supported by this representation. Because of time constraints imposed by the loss of funding, this capability was not ported to OMP II. To fully support the FBIS domain, this representation would need to be reintegrated into OMP.

5.1.2 Development of a Specification Language

The OMP prototypes are implemented completely in LISP. Adding new classes of resources, new types of activities, new heuristics (at any level), or any extension in the OMP capabilities requires the generation of additional LISP code. We recommend the development of a specification language that allows users to specify resources, activity structures, and heuristics using a high-level language. The users would then be able to tailor OMP to support their specific domain without the need for proficiency in LISP.

5.1.3 Specification of an Operations Interface

The main functions of the OMP II interface were to support development/debugging of the OMP system and to allow others to visualize the OMP multiphase approach to scheduling. The specification of an operational user interface, which would address how FBIS operators would use OMP in an operational environment, was scheduled for year three, but was dropped due to the restructuring of the task. While many of the existing interface features are transportable to an operational interface, it became clear during a series of in-progress demonstrations that operators would require additional support features and functions to be able to use OMP effectively.

5.1.4 Separation of Domain-Specific vs Domain-Independent Knowledge

OMP operates with a set of heuristics, which vary in terms of their generality. Heuristics can be categorized into those that are totally domain specific, those that are specific to a given type of problem but are general-purpose within that context, and those that are totally general-purpose. For example, a heuristic that enables a task to be moved from one resource to another would support most domains. Heuristics such as the load-assessment heuristics, which determine when OMP has developed significant levels of conflict to interrupt the loading and perform preliminary conflict resolution, are generally applicable to oversubscribed domains. Finally, heuristics such as those that enforce constraints concerning antenna hand-offs are domain specific. To evaluate the general applicability of OMP, an analysis is needed of the different types of heuristics and their generality, and an assessment of the dependence of OMP's performance on domain-specific heuristics.

5.1.5 Performance Characterization of OMP

A general analysis of OMP's performance is needed to characterize the OMP approach. It should address such questions as: How input sensitive are the schedules that OMP develops? Preliminary analyses indicate that, while the ordering of the specific tasks within a schedule is highly input sensitive, the amount of tasking accomplished by the schedule is not. How different is the resulting schedule when an alert is known a priori vs during execution? Can human schedulers easily improve upon the OMP-generated schedule? Under what conditions, if any, does running the optimizer result in a worse schedule? Where does OMP spend most of its computing time?

5.2 Research Issues

Advanced research into automated scheduling technology, based on the OMP approach, can follow several paths. The issues identified in the following subsections provide detail on several potential research areas.

5.2.1 Expanded Resource Structure

In its two incarnations, OMP has represented two types of resources: capacity and state. Capacity resources are limited to the number of tasks they can support at any given time, but once a resource is released, it is immediately available to support another task. OMP II represented antennas, relays, and translators using capacity resources. State resources can support a task only if they are in a given configuration, which can change with time. OMP I represented antennas as direction state resources. It is possible for a given resource to be both state and capacity.

In addition to these resource types, there is another, referred to as consumables. Consumable resources get used up by a task and are not available to support other tasks when the task is complete. Replenishables are a special type of consumable resource that can be replaced. An example of a consumable is the fuel on board a spacecraft such as Voyager (which cannot be refueled); an example of a replenishable is battery power, which gets drained and recharged throughout a spacecraft's life. OMP does not currently have the capability to support either type of resource.

5.2.2 Definition of Human Interaction Paradigm

Human interaction with automated schedulers is very limited. Usually the human is restricted to editing a schedule or submitting a task to be scheduled. There is no cooperative scheduling. The automated scheduler's interpretation of the human input usually falls on either extreme of a continuum. At one end, any changes the human makes in the schedule are regarded as sacrosanct and cannot be changed. At the other end, the scheduler accepts those inputs but can then disregard them, so that the operator input may simply be ignored. Neither of these two interaction paradigms is worthwhile for OMP-type scheduling.

With the OMP system, the control structure is instantiated through the multilevel control heuristics. It is envisioned that a human operator could perform the same functions as these heuristics. For example, the human operator should be able to designate a specific area of the schedule as a bottleneck and have the scheduler behave accordingly. Or, the human could set the focus state for the scheduler and have the scheduler focus its attentions on a specific point in the schedule. Or, the human could assign a task to a given resource. To prevent the scheduler from getting locked into a bad schedule due to these human scheduling actions, but also to ensure that the scheduler does not arbitrarily disregard them, a change in the interaction paradigm is needed somewhere in the middle of the continuum described above.

This interaction would vary the weight given to a specific human scheduling action based on the context of that action. When first enacted, the scheduler would regard that action as sacrosanct and would work around it. If that human scheduling action continues to cause serious problems with the schedule, the scheduler would become more willing to violate it. Finally, once a specified tolerance threshold is reached, the automated scheduler would violate that action and continue from that point to resolve any conflicts. There are several research issues involved in developing this type of interaction paradigm. They include identifying how the human operator can interact, when those interactions can occur, how the automated control structure interprets those interactions, and how the system deals with bad inputs.

5.2.3 Application of Concurrent Processing

Portions of the scheduling process, as described in the OMP iterative refinement approach, are inherently suitable for parallel processing. For example, the updating of resources or temporal regions could be done in parallel. Once bottleneck regions have been identified and interactions between different bottlenecks dissolved, each bottleneck can be resolved independently and is therefore a candidate for parallel processing. Identifying other opportunities for parallelism, managing the details of determining when interdependencies have been dissolved, providing distributed heuristic control, and controlling multiple access to the data structures are issues that must be addressed in the application of concurrent processing techniques to automated scheduling.

5.2.4 Identification of Heuristics To Aid in Optimization and Event Handling

The Optimization and Event-Handling Phases in the OMP II prototype are partially implemented. The high-level control structures and the assessment heuristics are only roughly implemented. Additional attention is required to determine how to set the focus state and

associated parameters, how to most effectively invoke the existing heuristics to create a better schedule, and what assessment functions are needed to support the Optimization Phase. Currently, OMP II hardwires which planning phase to reinvoke, based on the type of event. Additional heuristics are needed to support an immediate yes/no response to alert requests; assessment of the impact of a given event to determine the most appropriate phase in which to reinvoke scheduling; and handling of varying response times (e.g., yes/no response in n seconds, but x minutes during which the schedule can be improved).

5.2.5 Learning

An OMP-type automated scheduling system has two possibilities for learning: It can analyze the results of its own actions over extended periods of time, or it can analyze the types and results of human intervention in the scheduling system. These two types of learning can lead to extensions of knowledge bases supporting OMP's knowledge-intensive search and to the development and refinement of control heuristics. The application of case-based reasoning can assist in these learning processes. How to incorporate a learning system within OMP is a long-term research goal.

6.0 SUMMARY

The OMP task was successful in demonstrating the applicability of Artificial Intelligence technology to automated scheduling. The task resulted in significant accomplishments inspired by the development of the iterative refinement approach. The adoption of this approach, however, has added new questions and problems that need to be addressed through an ongoing, longer term scheduling research effort. However, even without addressing the extant research issues, the OMP approach can be reasonably extended to support operational domains. Therefore, the task succeeded both in demonstrating the applicability of state-of-the-art AI technology and in providing insight into the future research directions needed to further advance the concepts presented.

The research was performed by the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by the United States Department of Defense through an agreement with the National Aeronautics and Space Administration. The work was performed by members of the AI Group, which is chartered to perform advanced research on and facilitate transfer of new AI-based technology. The AI Group is part of the Computer Science and Applications Section 366 of the Informations Systems Division 360. In addition to the authors, the following people have contributed to OMP: D. Atkinson, L. Charest, R. Doyle, L. Falcone, K. Kandt, G. Martin, and H. Porta.

REFERENCES

[1] Biefeld, E., L. Cooper, and L. Falcone, *State of the Art in Planning, OMP Annual Report,* JPL Document D-5685 (internal document), Pasadena, California, Jet Propulsion Laboratory, April 12, 1988. (Included in this publication as Appendix F.)

[2] Dean, T., R. J. Firby, and D. Miller, "Hierarchical Planning Involving Deadlines, Travel Time and Resources," in *Computational Intelligence,* Ottawa, National Research Council of Canada 4(4), pp. 381–398, 1988.

[3] Vere, S. "Planning in Time: Windows and Durations for Activities and Goals," in *IEEE Transactions on Machine Intelligence,* No. 3, pp. 246–267, 1983.

[4] Tate, A., "A Review of Knowledge-Based Planning Techniques," in *Knowledge Engineering Review,* New York, Cambridge University Press, Vol. 1, No. 2, 1985.

[5] Fox, M., and S. Smith, "ISIS: A Knowledge-Based System for Factory Scheduling," *Expert Systems*, Medford, N.J., Learned Information, Vol. 1. pp. 25–49, 1984.

[6] Meleton, M. P., Jr., "OPT — Fantasy or Breakthrough?" *Production and Inventory Management,* Washington, D.C., American Production and Inventory Control Society, Vol. 27, No. 2, 1986.

[7] Smith, S., M. Fox, and P. S. Ow, "Constructing and Maintaining Detailed Production Plans: Investigations into the Development of Knowledge-Based Factory Scheduling Systems," *AI Magazine*, Menlo Park, Calif., American Association for Artificial Intelligence, Vol. 7, No. 4, 1986, pp. 45–61.

[8] Biefeld, E., L. Cooper, L. Falcone, and G. Martin, Operations Mission Planner Annual Report, JPL Document D-5685 (internal document), Jet Propulsion Laboratory, Pasadena, California, December 1, 1988.

APPENDIX A

OMP TECHNICAL REPORT

1.0 INTRODUCTION

Appendix A is the Technical Report for the Operations Mission Planner task. It presents detailed technical information on OMP implementation techniques that are not covered in depth in the main report or in the technical papers published on OMP and presented in Appendixes G, H, and I. The areas covered in this Appendix are Domain Representation and the Knowledge-Intensive Search.

2.0 OMP Domain Representation

There are two classes of knowledge represented in OMP. The first class describes the components of a schedule. This includes knowledge of the possible types of tasking and the various resources that make up a bureau. The second class of knowledge is represented in the various heuristics that direct the scheduling process. Representing this metaknowledge, which determines how to schedule, is a major part of OMP.

2.1 Task Representation

Within the first class of knowledge there are three distinct types of knowledge bases: the Task Expansions, the Resource Descriptions, and the Bureau Descriptions. A Task Expansion describes the possible set of activities that can satisfy a requested task. A Resource Description identifies the parameters used in describing an antenna and how an activity's step can reserve this type of resource. Finally, a Bureau Description describes the number and types of resources at a single bureau location.

In the FBIS scenario, a request for collection of a broadcast is represented as an input task. OMP uses the task-expansion knowledge to create a schedulable task. The two major components of a schedulable task are a task description and an activity tree. When OMP is started, one or more text files that specify the requested broadcast coverage are read. From these files OMP builds a task description for each request. A task description specifies the task type[1], name, priority, windowing, and any other parameters[2] needed to specify the task. Also contained within a task description is whether the task is currently scheduled or deleted, and information to interface the task with the graphical display.

While a task description specifies a request, an activity tree specifies exactly how a task is scheduled. For example, a task description specifies which antennas a broadcast may use, while an activity tree will specify which antenna the broadcast is using in the current schedule. Thus, the task description specifies the possible ways to schedule a task, while an activity tree specifies a unique configuration of a task.

The leaves in an activity tree are called steps. A step must contain a pointer to a resource, the amount of usage of that resource, and a temporal interval. When a task is scheduled, the

---

1 In the current scenario there is only one type of task, called a Broadcast; however, the system is set up to support multiple types of tasking. Each type would have its own set of parameters, scheduling actions, and tactics.

2 The set of parameters needed to describe a Broadcast is given in Appendix B.

resource timeline specified by the step is updated to indicate the task usage of the resource for the specified temporal interval. Thus, the steps in the task's activity tree are the mechanisms by which tasks interact with the resources.

The non-leaf nodes in an activity represent choices that simultaneously affect several steps. In a broadcast task, the particular processor is chosen in the root activity node. This means that all the processor steps in a broadcast must use the same processor. Other decisions, such as which antenna to use, will be chosen at the leaf (step) level in the activity tree. The parameters specified in a particular level of an activity tree are domain specific.

An activity node may vary the type or the number of its descendants. For example, in a broadcast activity tree the Antenna-Coverage activity node may contain one or more antenna steps. If the Antenna-Coverage node contains more than one antenna step, then the adjacent steps must temporally overlay each other by the antenna hand-off duration specified in the broadcast's task description. This is how a broadcast's activity-tree structure represents an antenna hand-off.

The OMP activity trees are similar to the goal expansion skeletons found in traditional planners. They allow OMP to represent a large variety of different types of tasking where there may exist many different ways of accomplishing a single task. The OMP activity-tree structure is unique in how the tasks interact indirectly through the resources and how the heuristics modify the activity trees during the scheduling process.

2.2 Resource Timeline Representation

Resources are represented in OMP as timelines. An OMP resource timeline monitors three different types of change in a resource over time. The first type is the state of the resource. In OMP II, this is the current usage of the resources and is graphically displayed as a histogram. In OMP I, the antenna's resource timelines state is the current direction of the antenna, which can vary between 0 and 360 degrees. The direction state is graphically displayed in OMP I as either a radarlike scope[3] or a flattened directional cylinder[4] timeline. Along with the time-varying state of the resource, the resource timelines track the steps which are requesting usage of the resource. This second type of information is used by the strategic heuristics to determine which tasks need to be modified, and it is graphically displayed as the resource Gantt charts. The last type of change tracked by a resource timeline is the chronology[5] of the resource. The chronology of a resource is used by the assessment heuristics to identify resource bottlenecks and is used by the control heuristics in deciding which region of the scheduling to focus on next.

The resource timelines are divided into eras, temporal regions long enough in duration[6] to be interesting to the top-level scheduling heuristics. The general architecture of OMP allows the

---

3 The scope display also presents the range information for an antenna but only displays the direction state for a particular moment in time.

4 The directional cylinder length represents time, and the circumference represents the direction of the antenna. The cylinder is "cut" down the length at 0–360 degrees and is flattened out into a temporal strip.

5 See Appendix F.

6 The exact value of this duration is "soft" in OMP and is domain specific.

tasks to be scheduled at a finer resolution than the duration of an era. The larger era resolution[7] eliminates the temporal "noise" from the top-level control and assessment heuristics. The resource chronology is kept at the era level and is updated by the strategic heuristics. This information tracks the *effort* the strategic heuristics put into a resource region.

2.3 Heuristic Knowledge Representation

There are three classes of scheduling metaknowledge in OMP. The first class is the dispatch heuristics which, in the current OMP architecture, is represented by *tactics* and *actions*. These dispatching heuristics along with the tactical search engine are responsible for modifying a scheduled task. The second class of metaknowledge includes the control heuristics which are implemented as a group of "strategic" search algorithms. These algorithms direct the search performed by the dispatch heuristics. The third class of scheduling metaknowledge includes the assessment heuristics. In OMP II, these heuristics are used to identify and classify resource bottlenecks. This information about the bottlenecks is used by the control heuristics to strategically direct the scheduling process.

The multiple levels of heuristic control are necessary to support OMP's scheduling approach, referred to as Knowledge-Intensive Search. In order to work effectively, the scheduling engine requires the knowledge encoded in the various levels of heuristics. To gain a better understanding of how the heuristics work, it is necessary to view them in the context of the Knowledge-Intensive Search process.

3.0 KNOWLEDGE-INTENSIVE SEARCH (KIS)

Knowledge-Intensive Search occurs at two levels: the tactical and strategic. Each level has distinct goals, although they operate cooperatively. The heuristics and type of knowledge are different at each level, although decisions made at one level affect the others. The tactical search engine performs the actual scheduling process of assigning resources to the tasks, but it does so in the context set by the strategic search engine. The following sections describe the tactical and strategic search engines in greater detail and identify how they support the concept of Knowledge-Intensive Search.

3.1 Tactical Search Engine

The goal of the Tactical Search Engine is to determine and implement the appropriate scheduling actions. It does this by manipulating the activity structure associated with the tasks. Instead of constructing a conflict-free activity expansion of a task, the OMP tactical search engine modifies a task's existing activity structure. The search engine tries various modifications in order to eliminate a "problem" that was identified by a control heuristic. The tactical search engine performs a depth-first search that either uses the first *acceptable* activity tree it finds or leaves the activity tree in its original state. The definitions of *acceptable*, *problem*, and the constraints (or context) of the search are set by the control heuristics. In most cases an acceptable

---

[7] Due to the tight schedule in developing OMP's two prototypes, this feature was never tested.

state is one that eliminates the conflict in the resource regions that the control heuristics have focused upon.

In the OMP II architecture, the activity-tree structure, actions, and tactics form the task expansion and the dispatch knowledge in the modification of a schedule. Instead of blindly trying different expansions of a task, the OMP tactical search modifies a task in direct response to an identified problem. Also, by setting the initial search context, the control knowledge can direct the tactical search. This is commonly done to restrict the search either to simple actions which have a large impact on the schedule (such as, try a different time window or delete a task) or to more complex actions (such as antenna hand-off). The simple actions usually have a large impact on the schedule and are thus used in the beginning of the search process to quickly rough out the schedule, while the more complex actions depend upon the schedule being relatively static and thus are of greater use at the end of the scheduling process. This allows OMP to progress from a search whose goal is to identify the scheduling bottlenecks to a search whose goal is to fit one more task into the existing schedule.

During the scheduling process, the control heuristics invoke a tactical search on a problem. A typical problem is a task's step that causes a resource oversubscription. The tactical search engine will first collect possible actions which may solve this problem. It then invokes an action which will modify the task's activity. After the task has been modified, a check is made for any new problems in the updated activity. A typical example of a new problem would be if the modification caused a new resource oversubscription that is unacceptable to the control heuristics. If no additional problems are found, then the search terminates successfully, leaving the activity in its updated form.

If a new problem is found, then the tactical search engine continues the search in a depth-first manner. It will collect and run additional actions in response to the new problem. If it cannot find any appropriate actions, then it will unwind to the previous search state and try a different action from the previous list of suggested actions. If none of the suggested actions leads to a problem-free solution, then the tactical search engine terminates unsuccessfully, and it leaves the activity in its original state.

Actions are a property of a node of a task's activity tree that are used to modify that node. For example, a *Change-processor* action is a property of the top-level node of a broadcast, while a *Change-antenna* action is a property of the broadcast's antenna-step. Thus, a broadcast must allocate the same processor for the whole task, while it may simultaneously allocate different antennas during the broadcast collection. An action heuristic modifies the activity node associated with the action. The action may invoke subactions on the activity node's descendent in order to propagate the effect of the action over the activity tree. Each action will post a corresponding unwind action on the unwind list. This allows the tactical search engine to rewind to the previous state. At the termination of a tactical search, the unwind actions are deleted. Thus, OMP only performs backtracking within a single invocation of the tactical search engine and does not perform global backtracking over the entire scheduling process.

The tactical search engine uses the tactical heuristics to suggest possible actions. Like the actions and unwind actions, the tactics are properties of an activity node. When a tactic is

invoked, it examines the problem description, the search context, and the current state of its associated node. If the particular tactic is applicable, it will suggest one or more actions that may solve this particular type of problem. Several different tactics on the same node may suggest the same action but with different parameters. For example, the *Try-a-different-temporal-window*, *Shift-temporal-after-the-problem-interval*, and *Shift-temporal-before-the-problem-interval* will all use the *Set-interval* action but will supply different intervals to the action.

When gathering possible actions, the tactical search engine starts with the task's step that is directly involved in the problem. The search engine invokes all the tactics for this activity node and collects the list of actions suggested by these tactics. The search engine then moves to the parent node and invokes all the tactics for the step's parent node. The search engine continues up the activity tree and collects all the possible actions for this type of problem given the current context[8] of the search. At the root node of the activity structure, the tactics may suggest deleting the task, while tactics at the *antenna step* level may suggest actions that try different antennas[9]. These suggested actions form a search level in the tactical engine search space.

When a tactic suggests an action, it also supplies any needed parameters and the appropriate context for this action. For example, to shift temporally to the right of an over-subscribed resource region the *Shift-right* tactic will suggest a *Set-interval* action, and the tactic will supply the end time of the oversubscribed region as a parameter to the *Set-interval* action. If this action is invoked, then the tactic will add *Shift-right* and *Set-resource* to the current search context. These additional contexts mean that any tactics run later in this branch of the search space will not consider moving the activity temporally to the left or trying an alternative resource to the one that the previous tactic used in determining where to shift. If the search engine rewinds over this action, then the addition to the contexts will be lost. Thus, the system may try to shift left or change to a different resource to avoid the resource oversubscription problem.

3.2 Strategic Search Engine

The strategic search engine is tasked with focusing the search. It provides the information to the tactical search engine that determines what types of actions are allowable and in what context to use those actions. The strategic search engine has several strategic heuristics that it uses.

The *Shuffle* strategic heuristics choose a resource conflict to focus on. The tasks involved in this conflict are modified using the tactical dispatch heuristics. The context of the tactical search restricts the search to using simple actions. The search tries to eliminate the conflict from the schedule, but it can create new conflicts in resource regions on which it has not worked

---

8 The initial search context is set by the control heuristics. However, tactics can add to the search context as described in the following paragraph.

9 In the current OMP scenario, the "processor step" has no tactic that will suggest trying a different processor. Since this decision must be made for the task as a whole, the "Try a different processor" tactics reside on the root node. Thus, the tactics are domain specific.

much. This can have the effect of actually making the overall schedule worse[10]. When the strategic heuristic chooses the next resource region, it will look first for conflicts which exist on resource regions it has previously worked on. This means that, if fixing one region means breaking another, the scheduling process will quickly get stuck in focusing first on one region then the other. When the heuristic notices that it has expended too much effort in a cycle[11], it invokes the assessment heuristics to identify the bottleneck.

The assessment heuristics search the resource timelines for regions of high scheduling effort. These regions are grouped into a bottleneck. The assessment heuristics then examine the bottleneck for type of resources, amount of oversubscription, potential amount of capacity, and temporal extent. Based on these factors, the assessment heuristics classify the bottleneck. This classification is then used by the control heuristics to direct the next phase in the scheduling process. For example, if the bottleneck is large in extent and is very oversubscribed, the next strategy will delete the task from the bottleneck. On the other hand, if the bottleneck is small in extent and the total demand in the bottleneck is close to its capacity, then the next strategy will perform a deep search in the bottleneck region and will try complex dispatch heuristics such as antenna hand-off in order to tightly pack the tasking in the bottleneck region.

4.0 SUMMARY

The demonstration of multiple classes of scheduling knowledge, the use of chronologies to identify scheduling bottlenecks, the classification of these bottlenecks in determining which type of scheduling heuristic to use, and the interleaving of finding and solving bottlenecks, were all major research objectives demonstrated in the two OMP prototypes. The purpose of developing these techniques is to show the feasibility of an automatic scheduler which can use the knowledge gained in trying to construct a schedule and which operates by continually modifying an existing schedule. These techniques should allow the construction of automatic and interactive schedulers which will be able to quickly and optimally[12] construct large and complex schedules. The same systems will also be able to maintain the schedule in a minimally disruptive manner.

---

10 This is actually desirable, because in order to substantially improve a schedule without making several changes simultaneously, it is many times necessary to temporarily cause additional conflicts.

11 Unlike OMP II, OMP I did not first focus on previous work on regions. Instead, it kept a more detailed chronology which was analyzed completely after the Resource Scheduling Phase. By having the strategic heuristics choose their focus on the partially built chronology, the system can find bottlenecks quicker and can interleave the Resource and Bottleneck Scheduling Phases.

12 By optimal we do not mean the theoretical maximum but a schedule which human experts will not be able to substantially improve.

APPENDIX B

FBIS SCENARIO

1.0 Introduction

Appendix B describes the test scenario used to demonstrate the OMP II Prototype. This scenario is a modification of the OMP scenario presented in the March 21, 1989, Scenario Document, which is included as an attachment to this Appendix. Detailed information on each of the resource types is presented in that attachment. Due to the reduction in the period of performance of the OMP task, the scenario was streamlined to provide a representative test case. The following sections describe the streamlining used in generating the scenario and the distributions used to generate the task parameters.

2.0 Scenario Description

A scenario consists of two parts, the set of resources available and the requested tasking. The following subsections describe the resources and tasking which define the OPM II test scenario.

2.1 Resources

The OMP II test scenario uses antenna, translator, and relay resources. Receiver resources are not modeled. There are four types of antenna resources: UHF, VHF, UHF/VHF, and FM antennas. Each antenna is modeled as a capacity resource (Section 5.2.1 of the OMP Final Report). Use of the antenna resources therefore is constrained only by the number of tasks assigned and the radio frequency of the antenna. The ability to model antennas as directional/state resources was not carried over from the OMP I Test Scenario.

There are three types of translators: French, German, and English. Each translator category is modeled as a capacity resource, which is analogous to having a given number of people available to do translating. Use of the translator resources is constrained by the number of tasks assigned and the language of translation.

There is only one type of relay resource. It is modeled as a capacity resource and its use is contrained by the number of tasks assigned and whether they are allowed to be relayed.

The resource configuration used in the OMP II Test Scenario is designed to saturate at approximately 300 tasks.

2.2 Tasking

The OMP II Test Scenario has 500 tasks which are requested during an eight-hour shift. These tasks are all modeled as basic broadcasts: tasks which occur at a given time for a given duration and which require an antenna-processor (translator and/or relay) assignment in order to be considered scheduled.

Tasks are essentially defined by their window(s) (start time, desired duration, minimally acceptable duration), radio frequency, language for translation and/or relay capability, priority[1], and their name. Additional information on the constraints associated with the types of scheduling actions (shrinking, gapping, hand-offs) can be derived from this information.

3.0 Distributions

Based on guidelines from the sponsor, there were some basic relationships between the different parameters used to describe the broadcasts. General guidance identified that there were very few high-priority tasks vs many low-priority tasks. In addition, the higher priority tasks tended to be of longer duration. The distribution table used to develop tasking which conforms to this guidance is given in Table B-1, Priority vs Duration Distribution. The numbers in the table correspond to the number of broadcasts per 100 that would have the given parameters.

Additional guidance stated that some tasks should be supportable by multiple resources. To incorporate this characteristic into the scenario, we developed a grid of resource assignment possibilities, described as resource pairs. Pairs of resources which could conceivably support the same tasks due to overlapping requirements were generated. For example, it was conceivable that a broadcast could have a frequency which allowed it to be received by both a VHF and an FM antenna; therefore, a new antenna resource was constructed which allowed the choice between a VHF or FM antenna. Similar reasoning allowed the pairing of a translator type with the relay resource. Therefore, although we have only four antenna types, there are actually seven antenna categories which can be used as task parameters. Similarly, the four processor types generate seven processor categories. The distributions generated to support setting the task parameters are given in Table B-2, Processor vs Antenna Distribution. The numbers in the table correspond to the number of broadcasts per 100 that would have the given parameters.

Given these parameter-distribution tables, a scenario was generated by creating 500 broadcasts which were randomly assigned parameters according to the distributions. Names were generated, and a pseudorandom process was used to assign the start times and number of windows for an individual broadcast. The maximum number of windows was limited to three (3), and the maximum duration was limited to ten time units (one hour and forty minutes).

[1]The priority scheme was changed to run from 1 to 10, rather than 1 to 100, to simplify the display.

TABLE B-1
PRIORITY vs DURATION DISTRIBUTION

Priority / Duration	1	2	3	4	5	6	7	8	9	10	Total
2	0	0	0	1	0	0	1	0	1	0	*3*
3	0	0	0	0	1	1	0	2	1	1	*6*
4	0	0	1	1	0	1	3	2	2	2	*12*
5	0	0	0	1	1	2	2	4	2	5	*17*
5-6	0	0	2	2	1	0	1	4	5	6	*21*
6-7	0	0	0	1	2	2	6	1	4	1	*17*
6-8	0	1	1	1	2	2	3	1	1	0	*12*
7-9	0	1	0	0	0	1	1	1	1	1	*6*
8-10	1	0	1	0	1	1	0	1	0	1	*6*
Total	*1*	*2*	*5*	*7*	*8*	*10*	*17*	*16*	*17*	*17*	***100***

TABLE B-2
PROCESSOR vs ANTENNA DISTRIBUTION

Processor / Antenna	French	French Relay	German	German Relay	English	English Relay	Relay	Total
UHF	2	1	1	1	0	2	1	*8*
UHF UHF/VHF	3	2	1	1	1	3	1	*12*
UHF/VHF	1	1	2	1	1	2	2	*10*
UHF/VHF VHF	2	2	2	3	1	3	3	*16*
VHF	2	2	2	2	1	4	2	*15*
VHF FM	4	2	3	3	3	2	4	*21*
FM	2	2	3	3	1	3	4	*18*
Total	*16*	*12*	*14*	*14*	*8*	*19*	*17*	***100***

Attachment to

Appendix B

Operations Mission Planner Scenario

1.0 Introduction

The purpose of this scenario is to test how the Operation Mission Planner (OMP) handles complex scheduling problems. While the test scenario is based on the Foreign Broadcast Information Service (FBIS) domain, the concepts demonstrated are relevant to several other scheduling domains.

This scenario explores several forms of complexity beyond the current FBIS scenario demonstrated in December, 1988. The first simply involves using a greater number of tasks and resources. This scenario will use approximately five hundred tasks and roughly one hundred resources. Another form of increased complexity involves having three fundamentally different types of resources, antennas, receivers, and processors, all of which must be configured in order to produce a schedule. Each of these different types of resources has different parameters, such as direction and language capabilities, which have to be handled differently. Besides having three fundamentally different types of resources, each resource type has many subtypes. For example, there are 13 different subtypes of antennas. Each subtype has a different set of capabilities.

The FBIS scenario is centered on an eight-hour shift at a single bureau. The bureau is assigned to collect a large set of broadcasts. Despite the multitude of resources located at a bureau, the number of tasks will far exceed the total resource capabilities.

In order to collect a broadcast, the bureau must configure an antenna, a receiver, and a processor. OMP must choose an antenna with the correct frequency band and range (sensitivity) and point the antenna in the correct direction. A receiver with the correct frequency band must be connected to the antenna, and a processor must also be assigned to this configuration. These resources, if properly configured, can also be used to simultaneously collect other broadcasts.

2.0 Tasking

The basic tasking in OMP is a request to collect a particular broadcast or a group of related broadcasts. There will be approximately 500 broadcasts in the FBIS scenario. Each task is assigned a priority. This priority is used to determine which subset of the total tasking will be scheduled. The individual broadcasts have six important parameters which are used in determining the resources necessary to collect the broadcast. These are: start time, duration, frequency, direction, range, and language.

The priority scheme used in the FBIS scenario ranges from 1 to 100, where 1 is the highest priority and 100 is the lowest priority. The priority scheme is absolute in the sense that collecting one more higher priority broadcast is more important than collecting any number of lower priority broadcasts. For example, collecting a single priority-12 broadcast is preferable to collecting a large number of priority-13 broadcasts. Table 1 gives the frequency of the tasking priorities.

Each broadcast has a start time and a duration. The start times are distributed over an eight-hour shift. The duration of a broadcast runs between 5 and 30 minutes and is skewed

towards the shorter duration tasks. It may be possible to shorten a broadcast collection by not collecting the last few minutes of the broadcast.

The frequency, range, and language of a broadcast determines which subset of the resources can possibly be used in the collection of this broadcast. The direction is used to determine if an antenna is pointed so that it can collect the broadcast. The use of these parameters will be described in Sections 3.0 through 5.0.

3.0 Antennas

There are four static characteristics of an antenna. These are the antenna's frequency band, range, angular reception, and rotation speed. The antenna's frequency band and range are used with the broadcast's frequency and distance to determine if it is possible for the antenna to be used in the collection of the broadcast. The antenna's angular reception and rotation speed are used with the broadcast's direction to determine the antenna's pointing requirements.

If a broadcast's frequency is within the frequency band of an antenna and the broadcast's distance is less than the range of the antenna, then this antenna can be used to collect the broadcast. While the antenna's range and the broadcast's distance are expressed in kilometers, this is really an expression of the sensitivity of the antenna and the broadcast signal strength. The broadcast distance can be calculated from the actual physical distance of the broadcast, the strength of the broadcast, and any known interference of the broadcast. The antenna's range is a measure of the antenna's sensitivity. By combining these concepts into a distance, one can easily show these attributes as a graphical scope display without sacrificing the accuracy of the scenario.

Each antenna has a time-dependent attribute that is the direction in which the antenna is pointing. OMP can change this direction at a rate less than or equal to the antenna rotation speed. The antenna can only collect broadcasts whose direction is within the antenna's direction, plus or minus half the angular reception of the antenna. The number and types of antennas are given in Table 2.

4.0 Receivers

Each receiver has a subband selector and two tuners. OMP can select the subband for each receiver from the receiver's list of legal subbands. Each receiver may collect two separate broadcasts, but each broadcast must be within the same subband. The number of receivers and the collectible subbands are given in Tables 3 and 4.

5.0 Processors

There are three different types of processors: recorders, relays, and translators. A recorder can process any one broadcast, while a relay can retransmit eight signals simultaneously. A translator can only process broadcasts that are in the same language as the translator. A translator can process one broadcast in the detail mode or two signals in the summary mode. Table 5 lists the types and numbers of the various processors.

Table 1. Task Priority Frequency

Priority Range	Task Frequency
1 – 7	20%
8 – 20	50%
21 – 100	30%

Table 2. Antennas

Type	Number	Frequency Band	Angular Reception	Range (Km)	Rotation Speed (Ang/Min)
TV	1	VHF	30	250	360/10
TV	2	VHF	70	200	360/10
TV	2	VHF	110	150	360/10
TV	1	UHF	30	250	360/10
TV	2	UHF	70	200	360/10
TV	1	UHF	110	150	360/10
TV	1	UHF	110	150	Fixed
Radio-AM	1	MF	110	200	360/10
Radio-AM	1	HF	110	200	360/10
Radio-AM	1	MF	30	300	360/10
Radio-AM	1	HF	30	400	360/10
Radio-FM	1	VHF	30	200	360/10
Radio-FM	1	VHF	110	200	Fixed
Radio-FM	2	VHF	180	50	360/10
FAX	2	UHF	15	200	360/10
Total:	**20**				

Table 3. Receivers

Band	Receiver Subbands	Number
MF	1/2	4
MF/HF	3/4	2
HF	4/5	2
HF/VHF	5/6	2
HF/VHF	3/4/5/6	4
VHF	7/8	3
VHF	8/9	3
VHF	9/10	3
VHF/UHF	10/11	3
VHF	7/8/9/10	4
VHF/UHF	8/9/10/11	3
UHF	12/13	3
UHF	13/14	3
UHF	14/15	4
UHF	12/13/14/15	4
UHF/SHF	14/15/16/16	3

Table 4. Subband Description

ID Number	Frequency Allocation
1	300 – 1600 KHz
2	1600 – 3000 KHz
3	3 – 6 MHz
4	6 – 20 MHz
5	20 – 45 MHz
6	45 – 55 MHz
7	55 – 65 MHz
8	65 – 100 MHz
9	100 – 160 MHz
10	160 – 200 MHz
11	200 – 300 MHz
13	300 – 500 MHz
13	500 – 700 MHz
14	700 – 900 MHz
15	900 – 1200 MHz
16	1200 – 2400 MHz
17	2400 – 3400 MHz

Table 5. Processors

Type	Number	Processing Volume
Equipment		
Recorder	6	1
Relay	8	8
Translators		
French	4	1 - Detail 2 - Summary
German	6	1 - Detail 2 - Summary
Spanish	2	1 - Detail 2 - Summary
Dutch	1	1 - Detail 2 - Summary

6.0 Alerts

The FBIS secnario will also be used to test how OMP responds to changing conditions. This is accomplished by submitting "alerts" to OMP. An alert is a high-priority task that is submitted to OMP during the execution of the schedule. This task is to be quickly added to the existing schedule. The FBIS scenario will contain about 50 alerts scattered over an eight-hour shift.

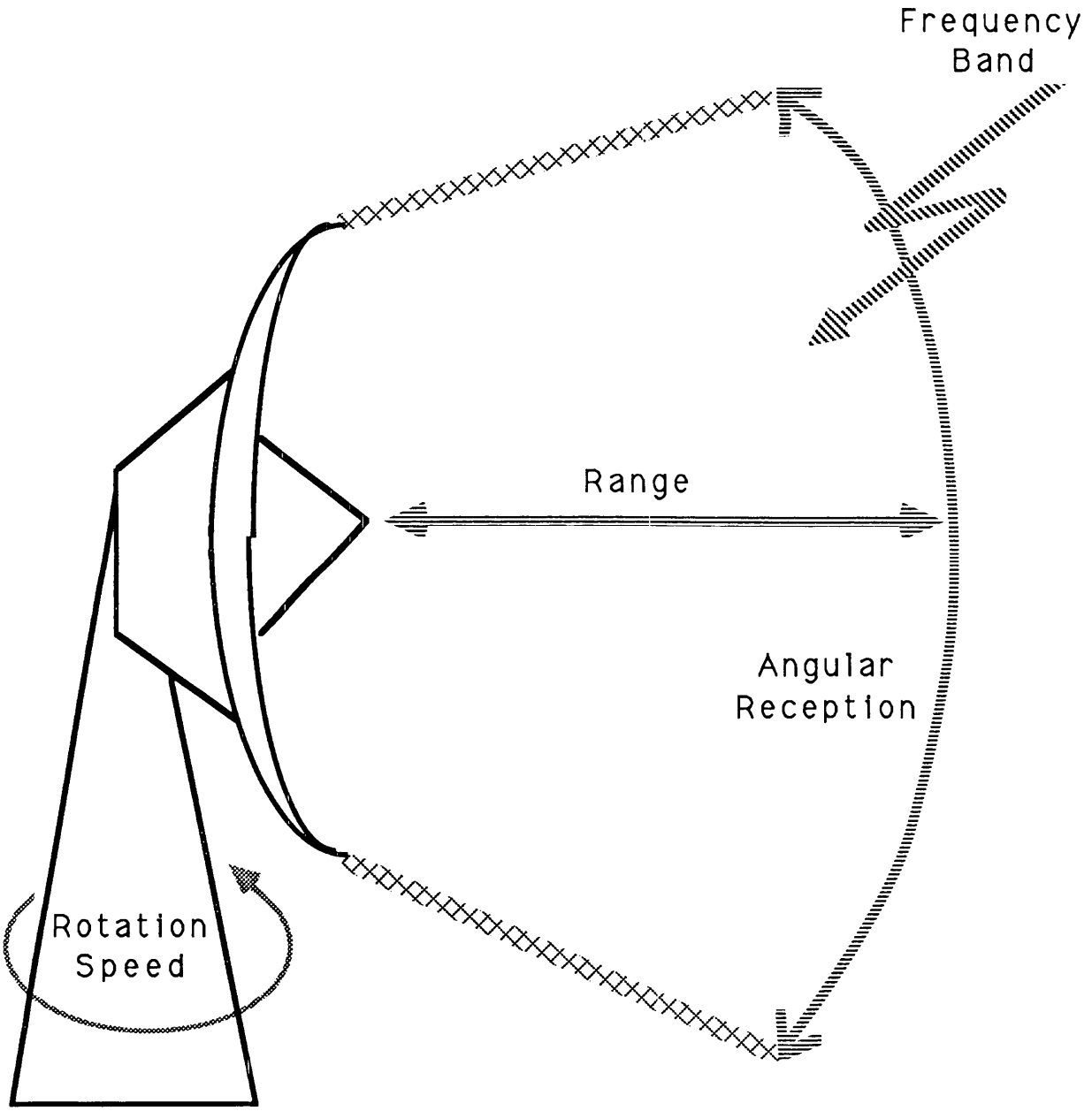


Figure 1. Antenna Characteristics

APPENDIX C

RESEARCH PLAN

1.0 INTRODUCTION

The Artificial Intelligence (AI) group of the Jet Propulsion Laboratory (JPL) will conduct research for the Central Intelligence Agency's Office of Research and Development (ORD) on AI contributions to resource allocation problems. The study shall be geared toward the scheduling process of the Foreign Broadcast Information Service (FBIS). Consisting of antennas, receivers, and processors, the FBIS is the system responsible for the optimal collection and dissemination of foreign broadcasts. To date, the process of FBIS scheduling has been performed manually and by operations research (OR) methods. Attempts to improve the scheduling process with classical methods have not been completely successful. Hence, by introducing AI techniques, we shall investigate the FBIS problem from a new perspective. This effort is called the Operations Mission Planner (OMP)

The Research Plan consists of several sections. After identifying important issues of the FBIS scheduling in the PROBLEM DESCRIPTION, we outline the BACKGROUND of AI planning and OR, which has focused upon a similar problem scenario. The APPROACH section focuses on new ways to handle FBIS planning. Here the concepts and implementation ideas behind minimal disruption are addressed. The TOOLS section gives a brief overview of the existing AI software pool at JPL. Then the STUDIES section identifies the categories of studies to be conducted in support of research. The previous sections will be synthesized in a schedule, and a chart will identify the accomplishment milestones, reviews, and prototype delivery date.

The central issue in resource allocation is planning. The FBIS is a flexible operation which possesses the capability to monitor many broadcasts efficiently. Planning becomes a critical issue as the FBIS becomes an oversubscribed system in a dynamically changing world. In order to be effective, FBIS must be able to respond quickly and efficiently to any changes that arise.

A scheduling system for the FBIS must meet three major requirements. It must respond to changes in a timely manner, it must maximize collection coverage, and in doing the above, it must minimize the disruption of ongoing collection efforts. While meeting these requirements, the scheduler must cope with the flexibility of the FBIS. To the scheduler, this flexibility means dealing with an astronomical number of possibilities.

A classic way to deal with the scheduling problem is to break the problem down into planning levels. First, a high-level schedule is created. For example, the monitoring tasks are assigned to the different bureaus. Then a detailed schedule is built from the high-level schedule, i.e., an expanded schedule is made for each bureau. The main drawback of this approach is the difficulty of shifting tasks among bureaus if the shifting becomes advantageous during low-level planning or shifting becomes necessary due to changes in the state of the world. The ability of the FBIS to cover a maximum number of targets would be lost in this case, and one could no longer capitalize on the FBIS's flexibility to adapt to the changing circumstances.

There are trade-offs between timeliness and disruption of the network, depending on how new goals are introduced into the system. Computerized systems are usually timely but result in

significant disruption of the schedule. Hand editing is minimally disruptive but requires much more effort and time.

Our approach will be to combine the knowledge representations, data representations, and problem solving from Artificial Intelligence with the heuristics and mathematical techniques from Operations Research. This hybrid system will use sophisticated methods which are minimally disruptive to enable a maximum number of broadcasts to be monitored. We will use advanced heuristics to limit the number of scheduling possibilities so that a solution can be found in a timely manner.

The result of this hybrid system will be an increase in the collection of broadcast material. This will be accomplished by efficiently employing the resources of the FBIS, minimizing the disruption to the network, and modeling states of the resources so that no surprises occur. The FBIS will then be able to adapt efficiently to changing conditions. Reassigning tasks to different bureaus to increase resource utilization will become a more feasible option under the new planner. The net effect will be the more productive operation of the Foreign Broadcast Information Service.

2.0 PROBLEM DESCRIPTION

The problem description is broken into two components. The first part will discuss the detailed requirements of the FBIS resource allocation task. The second part highlights the conceptual problems inherent in the resource allocation task.

2.1 FBIS's Detailed Problems

The Operations Mission Planner will efficiently allocate the resources of the FBIS. To accomplish this allocation, OMP must consider the broadcasts, bureaus, resources, and the changing conditions of the world.

The OMP will first decide which bureau can best be used to cover a given broadcast. Once assigned to a bureau, the broadcast will be allocated an antenna-receiver-processor combination. The OMP will make these assignments in a manner which causes the least disruption to the schedule.

For each bureau, there is a unique set of antennas, receivers, and processors. These resources have different characteristics. Thus, the OMP will have to match the frequency of the broadcast to the frequency coverage of the antenna and the receiver. The relative position and strength of the broadcast will determine the directionality and the sensitivity required from the antenna and the receiver. Other resources, such as human transcribers and relay capabilities, will be matched to the requirements of the broadcast. OMP will juggle these resources to obtain a schedule that maximizes the coverage of the high-priority broadcasts.

The OMP will produce plans covering the regular scheduled broadcasts. When an unanticipated, high-priority broadcast occurs, OMP will quickly assign the broadcast to a bureau and allocate the resources needed to monitor the broadcast. If the coverage of the high-priority

broadcast is in doubt (e.g., because the signal-to-noise ratio for the available antenna and receiver is too low), then the OMP may decide to assign the broadcast to more than one bureau to ensure complete coverage of the critical broadcast. These allocations must be done in a manner to minimize the disruption to the ongoing tasks of the bureau.

Not only do occurrences of unanticipated broadcasts impact the schedule, but also there are anomalous conditions to consider. These will include weather, jamming, and sunspots, which can increase the difficulty of receiving a broadcast. Other conditions, such as equipment failures, also impact the ability of the bureaus to carry out the current schedule. These anomalies will cause the OMP to dynamically reschedule the monitoring of broadcasts.

Besides assigning broadcasts to bureaus and allocating the resources at the bureaus, the OMP will track the important states of the FBIS's resources. For example, the OMP will track how many times the cables are wrapped around the various antennas. In another example, during an emergency broadcast, an antenna, A108, is preempted from its regular schedule. The OMP will check to see if antenna A108 can perform the reassignment. Assuming that this is possible, OMP will then check to see if the reassignment will cause problems after the antenna is returned to its regular schedule. OMP then notices that, during a regular scheduled task, after the emergency broadcast is over, antenna A108 will overextend its cables while repositioning from one regular broadcast to the next. This overextension was caused by the movement of the antenna during the emergency broadcast. OMP will then add to the schedule additional movement to unwrap the cable before the overextension occurs. OMP then notices that the unwrapping must occur after antenna A125 has finished its fifth task and has been repositioned for its sixth task of the day. Otherwise the two antennas will collide. All checking for the schedule will be performed in a totally automatic fashion.

The FBIS is an oversubscribed system. This means that not all the desired broadcasts can be monitored. The OMP system will decide which broadcasts should be dropped from the schedule. In doing this, OMP will consider the Agency directives and priorities for the coverage of broadcast material. While trying to maximize the coverage of broadcasts according to the above, OMP must also reserve enough resources to allow the search operators the flexibility they need to perform their duties.

2.2 Conceptual Problems

In this section we will present a scenario which highlights some of the problems inherent in the FBIS planning task. It will introduce and illustrate several concepts which will form the core of our research effort: exhaustive and heuristic search, nervous and non-nervous scheduling, multiple perspective scheduling, reactive vs long-range planning, and user modification of schedules.

> *It is 0800 at the Paris FBIS bureau. The operation plan for a new shift is beginning. The working antennas have been positioned to monitor their assigned broadcasts. One set of the medium-gain receivers is scheduled for periodic preventive maintenance. The*

bureau chief has completed the personnel schedule based on the operation plan for the day.

At 0805 Flight 457 out of Paris is hijacked to Iran. Monitoring the hijacked plane becomes the top priority for all bureaus. An updated plan is needed at 0805:01.

This type of planning can be viewed as classical job-shop scheduling. Job-shop scheduling is a well-studied field in mathematics. There are several known procedures that can find an optimal solution to this type of scheduling problem, but the trade-off is the amount of time that it takes to calculate an optimal solution.

Finding an optimal solution to a problem that is twice the size of another problem takes much longer than twice the time. For example, JPL's AI scheduler called DEVISER [1] takes about 10 minutes to run a small subset of the Voyager spacecraft scheduling problem. When a single new constraint is added, the computer time needed to solve the problem expands to between 4 and 14 centuries. This increase in time is due to the growth of the number of states that must be examined in order to find a solution. This explosive growth in the search space is known as combinatorial complexity.

A solution to the job-shop scheduling problem is not simple. Most real-world examples of job-shop scheduling are so large that optimal solutions may never be found. In fact, real-world examples usually have additional complexities, such as equipment failure and preemption of activities, which further complicate the scheduling problem.

The fundamental trade-off that we plan to use for this problem can be found in techniques from both AI and OR research [3] . The trade-off is to swap optimality for speed. Near-optimal solutions can be found by using planning heuristics. These heuristics, or rules of thumb, can be derived from human expert schedulers who have learned through experience how to quickly resolve certain situations that arise during plan construction. These heuristics can be used to limit the search space of the scheduler and thus reduce the time it takes to find a near-optimal solution.

The knowledge representations found in AI offer many opportunities to incorporate planning heuristics. The more effective planning heuristics are found in OR. It is hoped that, by combining these techniques, near-optimal solutions can be obtained in a reasonable amount of time.

At 0815 a new schedule is received from headquarters. This near-optimal plan includes coverage of the hijacking of flight 457 out of Paris. The bureau chief examines the new schedule. Flight 457 is to be monitored by a receiver that was previously assigned to cover a regularly scheduled news broadcast. A medium-gain receiver that was previously assigned for preventive maintenance is being reassigned to the task of covering the news broadcast. The bureau chief wonders why the medium-gain receiver was not

just assigned directly to Flight 457. It is identical to the one that was being used to monitor the regularly scheduled news broadcast. Why does the planner at headquarters insist on making life more difficult then it has to be?

The automatic planner at headquarters does not mean to make life more difficult. It just does not know any better. The automatic planner takes a set of goals and resources from which it generates a near-optimal schedule. When it assigns a receiver from a set of identical receivers it simply picks the first one in the list. Why not? They are all the same.

As the bureau chief walks down the hall, he continues to examine the new schedule. When he is just outside his office, he notices that not only has one medium-gain receiver been pulled off preventive maintenance, but all the medium-gain receivers were pulled off and assigned to other tasks. Instead of the medium-gain receivers being serviced, the new plan calls for a set of small-gain receivers to be serviced. This means that the bureau chief needs to reallocate the experts he has assigned for medium-gain receiver maintenance and replace some of them with his experts on the new small-gain receivers.

The automatic planner tries to optimize the plan to the best of its ability. It knows that it is better to do preventive maintenance on pieces of the same type of equipment at one time. When it has to assign an extra medium-gain receiver for monitoring the hijacking, the optimal set for maintenance becomes the small-gain receivers. The automatic planner has no idea what the plan was before the hijacking. It is constructing a new plan from scratch. Any small change in requests or resources may lead it to an entirely different solution.

In production management this is known as "nervous scheduling"[4]. Whenever the master planner is rerun to take into account changes in the goals or the resources, the new master plan is very different from the original. The philosophy that causes this is that "replanning is just planning again." This ignores the fact that the master plan is not an end unto itself. It is an integral part of a larger system. The master plan is an input into many other processes, such as the detailed implementation plan for which the bureau chief is responsible. The master plan usually goes through several reviews to check on its acceptability. Any change to the master plan causes additional work for the system as a whole to adapt to the new plan. The larger the change, the more work necessary to implement the new plan.

To solve this problem the planner must use the old plan as part of the planner's input. The planner must be able to iteratively modify an old schedule instead of constructing a new plan. Such a planner would have the additional advantage of being able to plan for new goals without the work of replanning for all the old goals. This should speed up the planner's reaction time.

Unlike the above example, sometimes several changes must be made to an existing plan in order to add a new goal. It may be necessary to bump receivers from one assignment to

another. The planner, however, cannot simply try all the possible reassignments. This problem is combinatorially explosive. The planner will need to rely on "plan surgery heuristics" to control the search of plan modifications.

One approach is "multiple perspective scheduling"[5]. Instead of just focusing on the constraints of the goals, the planner can focus on the resource bottlenecks. Many of the traditional planning heuristics are based on this concept. Because each plan can have different bottlenecks at different times, the planner must predict where the bottlenecks will occur. Two new production OR planners [6,7] accomplish this by first creating a preplan before the main planner runs. The preplanner sketches out the plan so that the resource bottlenecks can be identified. The information of the resource bottlenecks is used by the second, more sophisticated planner to focus the search. Both these planners, however, still have the problem of nervous scheduling.

Our approach will be to extend this two-pass technique. Instead of creating an initial plan sketch, the planner will use the original plan. The new goals will be overlaid on the old plan and the conflicts identified. This overloaded plan will then be used to find the resource bottlenecks. The knowledge of the conflicts and the resource bottlenecks will be used to focus the plan surgery heuristics. After the major conflicts of the new schedule have been eliminated, the planner will repeat this approach on the minor conflicts.

> *Despite all the problems, the monitoring of the hijacking went well. As the day is ending, one of the high-gain receivers burns out. Not much coverage is lost, but the replacement will take about 2 months. The planner at headquarters needs to be notified. What the bureau chief hopes for is a new schedule that is much like the current one for the next week. However, beyond a week's time span the bureau chief would expect the planner to produce a schedule that optimizes for the loss instead of trying to stay very close to the current schedule.*

This reflects the different expectations of reactive vs long-range planning. Reactive planning has to stay closer to the original schedule in order to let the organization adapt to the changes. It has to plan down to exact details. It must be capable of responding in a short time period. Fortunately, only the immediate future must be handled by the reactive planner.

In long-range planning, nervous planning is not so large a problem. The long-range plans are used to study basic capacity needs. The exact details are not necessary. However, the long-range planning does have to consider all the goals in the foreseeable future.

Actually, planning cannot be simply split into reactive and long-range planning. The range of planning styles between these extremes is a continuum. The planner must adjust its planning strategy in a more continuous manner. Ideally it would exhibit a large range of planning strategies so it could adapt to the current situation.

Early in the morning the Los Angeles bureau chief is awakened with a big shock. It measures about 8.2 on the Richter scale.

The central planner should note that this causes a very large change in the available resources. It does not make much sense to try to save the original plan. It is now time for some massive short-term planning. The strategy shifts to one of moving the high-priority tasks to other bureaus. For the bureaus that pick up the bulk of the load, the strategy is to cancel the lower priority goals to make room for the additional work load.

While a totally automatic planner is an attractive concept, the ability to hand-edit the schedule is still a necessity. The human in charge of the schedule may decide to make small changes to the schedule. This usually reflects a detailed knowledge about desirability of different scheduling compromises. When two goals are in direct conflict, the compromise between them may become political. The principals behind both of the goals may form a compromise that is not quite the one that the automatic planner suggested. Also, of course, the human may notice an improvement to the schedule that the automatic planner missed.

The planner should be able to modify the plan after a human has done some modification to the plan. However, the automatic planner should not simply restore the old plan, undoing what the human has just accomplished. The Plan-It system [8] allows the user to freeze any activity on the schedule so that the automatic planner cannot adjust it. In future planners a more intelligent approach is desirable. The user would be able to indicate to the planner either the conditions on which the editing depends or the desirability of maintaining the edited part of the schedule. The planner would then know when it is safe to change the involved activities, or just how desperate it needs to be before it tries to change this part of the schedule.

In summary, the issues of heuristics, non-nervous scheduling, and user modification of schedules are central to the development of an automated FBIS scheduler. These issues are all interrelated. The basic approach of the OMP is to integrate advanced representations with heuristics to control the search for new plans.

3.0 BACKGROUND DISCUSSION

In this section we describe previous work in automated planning. Emphasized are those planners which deal with issues relevant to the FBIS problem. These issues include reasoning about time, dynamic replanning, resource-focused scheduling, and resource conflict.

The automatic construction of plans to achieve goals has long been an area of interest in AI. Many programs have been written to explore different aspects of the planning problem. STRIPS [9], one of the earliest AI planners, laid the foundation for a host of AI planning research. Since then, research has addressed a variety of scheduling problems for different applications. Tate gives a historical survey of these AI planners in his paper on knowledge-based planning techniques [10].

The basic components of a general-purpose logical planner include a reasoning engine and a knowledge base. The reasoning engine is a program that manipulates facts in a given domain. The knowledge base contains descriptions of possible steps to be used in the plans.

The planner takes as input a description of the initial state of the world and a description of the goals (partial specifications of desired intermediate and final states of the world). Given the initial state, goals, and knowledge base of possible steps, the planner chooses steps which, if executed, would result in achievement of the goals. The same planning engine can be applied to different problem domains, given different knowledge bases.

3.1 Reasoning About Time

Some examples of general-purpose logical planners are NOAH [11], NONLIN [12], and DEVISER, which was developed at JPL. These planners are categorized as nonlinear, which means that they attempt to plan activities in parallel and order activities sequentially only when such ordering is necessary. Such a planner must know what facts in each part of its tentative plan are required to hold true to enable the execution of steps in other parts and must know the times at which facts become true or become false. NOAH was the first nonlinear planner, and NONLIN and DEVISER implemented capabilities for dealing with more aspects of the changes of facts over time.

FORBIN [13] is another general-purpose planner. It resulted from an attempt to combine many AI planning approaches into one planner. It also reasons about time. FORBIN uses the Time Map Manager [14] to store facts and partial knowledge about the times at which the facts hold. The Time Map Manager is a separate program that is meant to be incorporated in different expert systems.

3.2 Dynamic Replanning

One assumption that is often made about the application of an automatic planner is that it will be used in situations in which the goals and possible actions are stable. The planner is given a set of goals and generates a plan to achieve them. It is assumed that execution of this plan does not interfere with execution of any plan that was or will be constructed to achieve another set of goals.

In more hectic domains, including the FBIS problem, new goals are added at random times, even during the execution of existing plans. In this case, contrary to the assumption of goal stability, the planner, human or mechanical, must replan so as to achieve the new goals, beginning with the state of the world as it is at some stage of partial execution of the existing plan. The planner must decide what parts of the existing plan will persist in the new plan and what will be deleted or changed. The SWITCH/Runaround program was developed at JPL to replan in just such a situation [15]. It replans nervously, preserving only those parts of the old plan that could not possibly be changed and replanning everything else from scratch.

More difficult than the problem of dynamic replanning for changing goals is that of dynamic replanning in the face of dynamic changes to the state of the world and to the

capabilities of the executors of the plans. We refer to this kind of replanning as discrepancy replanning, because it must be done when there is a discrepancy between the state of the world predicted by the existing plan and the state of the world as it is observed. In the FBIS problem, discrepancies might include equipment failure, personnel absence, and changes of broadcast schedules at the sources.

Another automatic planner with replanning capability is Wilkins's SIPE [16, 17]. SIPE does some discrepancy replanning: It can replan to recover from execution errors. Its replanning is likely to be less nervous than SWITCH/Runaround's because it replans by modifying, deleting, and rearranging individual steps of the existing plan. We refer to such alterations as plan surgery.

3.3 Resource-Focused Scheduling

Reasoning about resources can be effective in directing a planner. The ISIS program [18] provided a rich set of constraint representations for the job-shop scheduling domain. While ISIS used a form of constraint satisfying, it did not make use of resource-focused scheduling. The OPIS project tries to combine the ISIS approach and resource-focused scheduling. The planner first tries to identify the resource bottlenecks and use this information to focus the search. The OPIS project claims a significant improvement over the ISIS system.

At JPL, the Plan-It program developed representations for resources. These resources were modeled after the mission timelines used for Voyager spacecraft scheduling. RALPH is a heuristic OR planner that is being developed at JPL to do the scheduling of the Deep Space Network (DSN). The DSN is a collection of antennas that are used to communicate with NASA deep space probes. RALPH is a two-pass scheduler whose first pass does a heuristic allocation and whose second pass is a dynamic programming routine. The first pass produces a resource likelihood profile that is used by the second pass.

3.4 Resource Conflict

In the FBIS domain, a typical set of goals for a plan will contain more goals than can be satisfied with the available resources. A program to solve planning problems of this kind should produce a plan for achieving at least some of the goals. It should aim to produce a plan that maximizes the number of goals achieved, or some other measure of degree of satisfaction of the goals, even though it fails to satisfy the set of goals completely. SWITCH/Runaround can skip goals [19].

Instead of deleting goals to "solve" a planning problem in which not all goals can be achieved, the planner might return one or more inconsistent plans which "achieve" all the goals by overloading resources. People in authority could then see which goals were conflicting with one another and decide which ones to delete. Plan-It has the ability to develop, modify, and return plans which include resource oversubscriptions.

3.5 Conclusion

Reasoning about time, dynamic replanning, resource-focused scheduling, and resource conflict are important issues to consider in the design of the automated OMP. Each issue has been at least partially addressed by previous automated planners.

4.0 APPROACH

To accommodate the need for a non-nervous scheduler, we shall implement a special-purpose planner that concentrates on the FBIS problem. The planner will use a technique called plan surgery. Several research issues stem from this approach. Representation, operation heuristics, tactical control and user interface will be examined. We will build a plan surgery software prototype to serve as a testbed for investigating these questions. The specific features which we expect our research-and-development effort to enable are:

(1) Plan Surgery
  (a) Operation Heuristics
  (b) Goal Representation
  (c) Tactical Control
(2) Direct Translation Interface

Plan surgery is the act of performing operations on a plan in order to add small changes without provoking a large disruption of the overall schedule. In brief, the approach will let the replanning begin by entering the desired goal change into the plan. The iterative replanner procedure will first produce a suboptimal and inconsistent plan. As replanning proceeds, plan surgery will change the plan until it becomes consistent and near optimal. Guidelines of iterative replanning will come from a set of operative heuristics.

Many planners such as DEVISER and SWITCH have a very limited repertoire of surgical operations. They often encounter situations in which a seemingly small change to the plan requires a tremendous effort on the part of the planner, or is utterly beyond the planner's ability. For example, if the existing plan already specifies the position of an antenna at 0200 hours and a move from that position to another for 0400, SWITCH probably could not insert a move to another position at 0300. The reason for this is that SWITCH's representation of the act of moving to the 0400 position would already require that move to start from the 0200 position, and a move to another position at 0300 would violate that requirement. SWITCH has no easy way to disconnect the 0400 move from one starting position and reconnect it to the other. A planner with more surgical operations would be able to insert an intermediate move.

This approach requires an extensive repertoire of surgical operations, control heuristics, knowledge base and proper representation. We will discuss each of these in detail.

4.1 Representation

We must first examine the representations necessary to support the plan surgery approach. We want to represent plans in a manner that will make plan surgery operations easy to

implement. (Of course, the representation must also capture all the essential nuances of the problem domain.) We will take advantage of the nature of the problem domain to build this plan representation. It will reflect the nature of change propagation links (or couplings) and goals.

To understand how a change propagates throughout the schedule, we will define a linking system. The link types fall into two categories, loose couplings and tight couplings. One useful aspect of the problem domain is that many activities are loosely coupled, i.e., a change to one activity does not necessarily have a large-scale effect on the rest of the plan. For example, if the goal that led to planning to have the antenna in a certain orientation at 0200 is deleted and replaced by a goal requiring a different orientation, it does not invalidate the move to the 0400 position; it only means that the move starts from a different starting position. We will categorize different types of loose couplings that exist in this problem and implement plan surgery operations for each type.

The mechanism of the loose coupling corresponds to the resources in the problem. For example, the goals in the above example, of receptions at 0200 and 0400, interact with each other because they share the same antenna. Achievement of the first goal leaves the antenna pointed in one direction. The planner must include a redirection of the antenna for the 0400 goal. The loose coupling can be described in terms of allocation of the antenna as a resource. Our approach will be to cover all the loose couplings in terms of the resources.

Tight couplings, on the other hand, are relationships which cause a large-scale and direct effect on the rest of the activity. There are some tight couplings evident at some levels of detail in the plan. Typically, an activity of receiving a broadcast will expand into steps of setup, reception, and teardown. These steps are tightly coupled. If the start time of one step changes, then in a single planning operation, the start and finish times of all steps should be updated. This will be implemented by a classical constraint-propagation method, such as that found in ISIS, MOLGEN [20, 21], or the Time Map Manager.

4.2 Goal Representation

Now we consider the goal representation. The plan will be initialized by translating the goals into demands on resources. The replanning process consists of two steps:

(1) Goal Translation into Resources
(2) Resource Management

A goal of covering a broadcast will be turned into a sequence of steps and resource demands by a process of goal expansion similar to that of FORBIN. For instance, a goal requiring a human to monitor a broadcast will expand into steps of setup and listening. At almost the same time as the expansion, an antenna, receiver, and operator will be chosen for this monitoring task.

It will most likely be the case that, after goal expansion, the plan will be inconsistent in some way. For instance, some resource, such as a specific antenna, will have become oversubscribed, required to point to more than one source at a time or to be undergoing

reorientation at the same time that it is supposed to be fixed on some source. After this goal translation, there will follow a reassignment process. In such a case, plan surgery operations will be used to reassign some activity to another antenna or to make a more drastic adjustment, such as dropping some activity entirely along with its corresponding goal.

4.3 Operation Heuristics

Next, we must examine how to operate on the above defined representation. The sequence of activity manipulation is considered. Plan surgery operations provide the planner's capability to make small (as well as large) changes to the plan. The planner must also have a control structure that examines the partially constructed plan and decides what change to make next, i.e., what operation should be applied to what part of the plan. For example, if the plan shows that all antennas would be oversubscribed at some time, one possible change is to start to delete activities from the plan. Another possible change might be to try to reassign some activities to different antennas so as to save some setup time on some antenna and use that antenna for reception during the freed time. This sequence of activity manipulation translates to a set of operation heuristics. We will discuss the search heuristics that confine the search space and domain-specific heuristics that concentrate on the FBIS problem. Both sets will concentrate on optimality and avoid endless loop situations. Below we discuss these heuristics and their implementation concepts.

Because several operations might be applicable to the plan at the same stage, it is necessary to decide which one to try first. The problem of plan surgery becomes a search problem and can even be approached as an exhaustive search problem. When searching among multiple alternative operations, it is desirable to try first the one that is most likely to lead to success. In other words, each choice of a plan surgery operation should be made according to some plan surgery heuristic.

We will implement several sets of plan surgery heuristics to control the action of plan surgery operations. Different sets will be active at different times during plan generation. Some sets may be composed of heuristics that favor operations that lead to a solution quickly. Among the operations most conducive to fast plan generation are those that delete goals and activities. Other sets of heuristics may be composed of heuristics that tend to lead to optimal plans. The heuristics in these sets would tend not to favor operations that delete goals. Other sets of heuristics may favor operations that make changes that are least nervous.

We will determine and employ domain-specific heuristics as well as more general-purpose ones. One heuristic that applies to a wide range of resource allocation problems is that of making a rough schedule to determine the resource bottlenecks and letting them drive the refinement of the schedule. Some planners that use this procedure are OPT and RALPH.

In the first stage of development, the heuristics will be similar to standard "If (situation) then (action)" rules. In the future, OR routines may be included as heuristics.

It is easy to imagine a naïve plan surgery control that would enter an infinite loop of repetitions of the same operations. For example, it is reasonable to expect that somewhere in the sets of heuristics there would be two heuristics, such as the following:

IF some resource is oversubscribed at some time,
THEN choose one of the activities using that resource at that time and delete the goal which expanded into that activity.

If the plan is consistent, AND if some goals have been deleted,
THEN select a deleted goal and try to restore it to the plan.

(The first of these heuristics would be used to reduce inconsistency in a plan under construction. The second would be used to increase optimality, i.e., to increase the number of goals achieved.) Consider a plan that is almost consistent, but in which just one activity oversubscribes one resource. In this case the first heuristic might be applied. Then one goal would be deleted and the plan would become consistent. Then the second heuristic might be applied, restoring the prior situation in which one resource is oversubscribed by one activity. Then the first heuristic might be applied again, restoring the second situation in which the plan is consistent and there is a deleted goal. Then the second heuristic might be applied again. We have an endless loop. Unless something eventually interrupted the planner, it would delete and restore the same goal repeatedly. We will make the planner control smart enough to interrupt itself out of any infinite loops, including those that are much more subtle than the above two-step loop.

The criteria by which the planner decides that it may be in an infinite loop will, of necessity, be over-conservative. It is well-known that there is no general test, applicable to every computer program and choice of data values, for determining whether the given program will halt or cycle forever (see, for example, [22]). In claiming that the planner will interrupt itself out of any infinite loops, we are not claiming that we will have solved the halting problem.

The conservative test for infinite looping will interrupt some searches that take a long time. It will provide some control over the amount of time that the planner runs and some control over the extent of changes the planner makes to the existing plan. However, it may prevent the planner from finding some consistent plans, possibly including the optimal plan.

Plan surgery, properly implemented, will contribute to the planner's ability to provide near-optimal schedules in a timely fashion, as well as enable non-nervous replanning. The ability to make small changes to the plan with small effort can greatly speed up the search for optimal plans, if the changes are chosen carefully. On the other hand, the more surgery operations the planner has available, the more redundant paths there are through the search space. So, if the changes are not chosen carefully, increased plan surgery capability can actually make the search take longer. The secret to success is in the control.

4.4 Tactical Control

We now further consider how to confine the search space. The plan surgery operations can modify any facet of the plan. They can intelligently update the plan data base and notify the

system of any resource or constraint violations. The plan surgery heuristics can invoke these plan surgery operations. The plan surgery heuristics are domain-specific tricks that can be used to tweak the plan. The heuristics are miniature searches using AI or OR techniques. There are also pattern matchers that identify the resource bottlenecks. This information will be used to focus the plan surgery heuristics. These are all ingredients of tactical control. This tactical control will trigger and set up the focus for the plan surgery heuristics. Below, we examine the facets behind tactical control to avoid endless loops, to confine the search space, and to allow fine tuning.

The tactical control embodies the scheduling metaknowledge. This will include the control logic that causes the planner to focus first on the resource bottlenecks. If the schedule is oversubscribed, this metaknowledge will trigger the heuristics that delete low-priority goals. If this does not sufficiently reduce the level of oversubscription of the resources, then it will delete some of the higher priority goals. This type of planning metaknowledge will reduce the search space of the planner and thus help to increase its speed.

The tactical control must also watch over the execution of the plan surgery heuristics. Since these heuristics are a large collection of domain-specific strategies, it is almost impossible to keep them from getting into endless loops. In a problem that seems almost solvable, the plan surgery heuristics may just keep trying to fix the problem. This could result in one of two problems. One is the problem of getting stuck in an endless loop. The classical technique for solving this problem is to introduce a rigorous search strategy. But such a strategy would defeat the power of having a large set of differing plan surgery heuristics.

Another problem that the plan surgery heuristics can get into is to slowly keep expanding the area of the schedule that they are working on. This will lead the planner back into nervous scheduling. The tactical control will set boundaries for the plan surgery strategy. The plan surgery heuristics could suggest that the tactical control expand the area of the search. If the current search fails to find a solution to the current problem, then the tactical control could use these suggestions in order to expand the area of the search. Or, the tactical control could decide that this would either cause too great a change in the schedule or simply take too long. In this case the tactical control would either trigger a set of more powerful but less than optimal plan surgery heuristics or simply abandon the current goal. Abandoning the current goal is really just a very powerful but not an optimal heuristic.

Once the tactical control is in place, it should offer other advantages. A major advantage will be fine-tuning the control of the plan surgery heuristics. For example, suppose that there is a set of plan surgery heuristics that perform a dynamic programming optimization on matching up the receivers and antennas to the set of targeted broadcasts. It is noticed that, if the heuristic spends most of its time reallocating either just the receivers or just the antennas, instead of reallocating both the receivers and the antennas, then the final result is not much of an improvement. Instead of letting the heuristic run to completion, the tactical control could use this domain-specific knowledge to terminate the heuristic earlier. The tactical control could also favor different levels of optimization versus minimal plan disruption for different resources at the different bureaus. For a hypothetical example, it is much easier to physically change the receivers attached to an antenna at the Los Angeles bureau than at the Paris bureau. This is due

to the new computerized switching just installed at the Los Angeles bureau. On the other hand, due to the difference in sizes of the antennas, the Paris bureau can respond much more quickly to a reassignment of antenna positions. The tactical control will accommodate this type of planning metaknowledge without having to resort to building different plan surgery heuristics for the different bureaus.

The plan surgery heuristics are the necessary pieces to add scheduling knowledge to the planner. This knowledge will allow the planner to effectively perform plan surgery. Since this scheduling knowledge is diverse and represents a variety of search strategies, there has to be a high-level control to guarantee the proper operation of the planner. This control knowledge will have to decide when a particular scheduling strategy is appropriate and when the strategy is no longer useful. It will allow for fine-tuning and avoiding endless loops. This control will have to be adaptable to a large variety of scheduling strategies. This is the purpose for the tactical control module.

4.5 Interface

An integral part of the automatic planner is the user interface. There are several major goals of the user interface. Its purposes include simple control of the planner: starting and stopping it, submitting goals, and approving the final answer. Other purposes are related to monitoring the planner while it is running. These include checking the states of the activities and the resources, both graphically and in menus. This ability also contributes to the user's understanding of how the planner approaches a conflict in the schedule, so new heuristics can be created and checked. Still another purpose is to allow the user to modify the schedule by hand.

The planner needs to display the current state of the plan and the current state of the resources. This allows the user to tell not only what goals have been achieved but also how tightly packed the schedule is. This information is needed so the user can edit the schedule by hand. The information is also needed for understanding how well the planner works. Given this type of display, the user can make intelligent choices while editing the schedule. Also, the user needs to understand how the planner reacts if domain-specific heuristics are to be added and maintained.

At any time, the user should be able to edit any activity or resource. This allows the user to input any new changes in the state of the FBIS. If a sunspot starts to interfere with broadcasts, the user needs to adjust the capabilities of the antenna resource. The impact of this change on the state of the plan should be immediately displayed. As the planner tries to recover, the display needs to show the modifications to the goals and how the planned goals affect the resources of the FBIS. In addition, if an activity does not go as planned, the planned activity must be edited so that the planner can calculate the impact on the ongoing plan.

As the planner operates, it will necessarily make compromises between different goals. The user may wish to tweak these compromises. This can be done by directly editing the activities. The planner must still be able to operate on the plan after the operator has modified these activities. The real problem will be keeping the automatic planner from needlessly readjusting the compromises. This could be done by using an activity freeze switch. This allows

the user to freeze any activity so the automatic planner cannot modify it. This is, however, an extreme solution, and an intermediate control between all or nothing should be investigated.

The interface will be invaluable, inasmuch as it shows how the planner is operating. This knowledge will be absolutely necessary if new planning heuristics are to be added to the system. By watching the planner operate, its weak points can be discovered. These points are where the planner either does not find a good solution or wanders around too much before it finds the solution. The planner interface should provide the user with the ability to display the same abstractions of the schedule as the heuristics use for input. This will include examining the classifications the resource bottleneck pattern matchers make and the focus state that the tactical control sets. This allows the user to see how the planner is viewing the planning process. The user will need direct control of the actions that the heuristics can use. This would include invoking the plan surgery operations and setting the focus state of the tactical controller. In this way the user can test out the effects of a proposed heuristic without having to actually write it.

During the operation of the automatic planner, the user can also use the control over the tactical focus state to direct the planner. The user could tell the automatic planner which part of the schedule should be worked on next. In addition, the user could force the planner to terminate a search that the operator thought was getting nowhere and start a new search.

The basic approach to the interface is to directly display the various structures on the screen. This is the same approach that was taken and was found quite successful in the Plan-It project. Each resource will be plotted as a timeline showing the state of the resource usage. Activities will be plotted on timelines that allow the user to graphically modify them. The activities will also be displayable through the use of menus. As the activities are moved, the graphic and menu displays of the activities will be updated. The effects of the activities on the resources will also be directly shown. Since a user would be overwhelmed by the total amount of the displayable information, the actual contents of the screen will be under the user's control. The user will be able to zoom in on the timelines. The user will be able to pan to any section of timeline displays. The selection of resource and activity displays that are currently on the screen will be controllable by the user. The user could add or delete or rearrange the displays on the screen at any time. The first prototypes of this display will be built using the current Plan-It graphic code. The graphic code will be adapted to display the OMP planner data structures, and color coding will be added to highlight important information on the screen.

An important aspect in the use of the automatic planner is that it should enable the human expert scheduler to intuitively grasp what the automatic planner is trying to accomplish. This means that the tool must not only be user-friendly, but must also be user-natural. Otherwise, the tool can only be effective used in a batch mode, no matter how fancy the graphic displays are.

To implement an effective non-nervous scheduler, we shall investigate the areas of representation, operation heuristics, tactical control and user interface. New, theoretical results shall be implemented if time permits. Otherwise they will be identified and outlined for future research.

5.0 TOOLS

The software development of the prototypes will be based on tools available at JPL. These tools can include STAR*TOOL, Plan-It, SWITCH/Runaround, Time Map Manager, FORBIN, and BB1. STAR*TOOL, Plan-It, and SWITCH/Runaround were all developed for JPL projects. Time Map Manager, FORBIN, and BB1 were developed at Brown University, Yale University, and the Stanford AI Laboratory, respectively.

The Time Map Manager is a general-purpose truth maintenance data base that maintains temporal relationships on a collection of facts. It allows a fact to be inserted into a temporal data base. The system enforces consistency for all the facts. Unlike a typical truth maintenance system, the Time Map Manager allows for temporal interactions of the facts.

Conceptually, each class of resource will need its own set of plan surgery operators and its own set of bottleneck pattern matchers. What is desired is to implement these protocols using the Time Map Manager. This will allow for a very sophisticated and efficient temporal data base to underlie the resource protocols.

FORBIN is an advanced AI planner. It was built using the Time Map Manager. There are several modules of FORBIN that we plan to use. One is the goal expansion module. FORBIN uses a goal library approach similar to that of MOLGEN. Controlling the goal expansion module is a set of heuristics which allow for the intelligent guessing of which expansion to try.

BB1 is a blackboard system to study the effects of control metaknowledge on problem solving [23] . Most BB1 systems, such as PROTEAN [24], use BB1 as a shell for both the metacontrol and a general-purpose blackboard for representing the problem. We wish to use the Time Map Manager for the low-level problem representation. However, since we have the code for BB1, we wish to try its meta-blackboard for representing the tactical control of the OMP.

Plan-It is a JPL program which contains a user-friendly graphical interface. The graphical interface to Plan-It has been separated from the rest of the program by NASA Ames Research Center. The generic research interface will allow for the display and editing of schedules independently of the underlying representations. This interface needs to be extended to include color coding of the resource and activity timelines. This will be the basis for the first prototype user interface.

STAR*TOOL [25] is a set of software tools for designing and building reasoning engines. STAR*TOOL stands for the Tool Environment and Language for Expert System Investigation and Synthesis and was developed at JPL. The STAR*TOOL tool kit includes many of the important AI paradigms. These tools may be employed independently or in any combination. The tools can be directly used in Common LISP. This allows the programmer complete flexibility in using these tools. STAR*TOOL can be used to help construct any module of the planner that is not taken from existing projects. This could include the plan surgery heuristics. STAR*TOOL can also be used to help glue the various modules that will be taken from existing code. This should lower the integration time and lead to a more flexible prototype.

The initial prototype will concentrate on dynamic replanning. The SWITCH/Runaround System can be used to generate the initial plan. In later prototypes the dynamic replanner will be expanded to include the generation of the initial schedule.

6.0 STUDIES

A series of studies shall be conducted throughout the OMP development. The studies will cover FBIS application, evaluation criteria, complexity, manual override of the automatic planner, and centralized vs decentralized allocation of resources.

6.1 Functional Requirements Document

The functional requirements document shall outline FBIS application requirements for the automated planning capability. To define the constraint prioritization, resource limitations, and anomalies, a detailed study of the FBIS application will be performed. The document is intended to clarify FBIS problem definition and approach.

6.2 Other Studies

6.2.1 Evaluation Criteria

Criteria will be established for measuring how well the automated planner meets objectives. Possible criteria include numerical measures of plan value, non-nervousness, and speed; human scheduler appraisals of plans produced by the automated planner; and ratings of the ease and usefulness of the interface.

6.2.2 Complexity Analysis

The computational complexities of the FBIS resource allocation problem, and of the automatic OMP program, will be analyzed. (The computational complexity of a problem or algorithm is a worst-case bound on the computation time required to solve or run it, as a function of some measure of the size of the input.)

6.2.3 Centralized vs Decentralized Resource Allocation

A study to evaluate the trade-offs between centralized and decentralized allocation of resources will examine the level of plan detail presentation to a bureau. Centralized resource allocation will disseminate a detailed plan to each bureau. Decentralized resource allocation assigns a list of responsibilities and constraints to each bureau. Centralized allocation offers top-level control of detail, but decentralized allocation offers bureau flexibility. How does each perform under dynamic scenarios? Which system performs best during disasters? Is there a combination which offers promise?

6.2.4 Manual Override

A study to assess the impacts of manually overriding an autonomous planner will be performed. How should the planner be informed of an act of manual overriding? What adjustments should the planner make to other parts of the plan after a manual override?

7.0 YEAR-BY-YEAR OBJECTIVES

We expect that a planner with the capabilities described in the APPROACH section above could be built in three years. At the end of each year we propose to demonstrate the accomplishments to date.

In the first year, we will concentrate on non-nervous replanning from an existing plan when a new goal is added. We will model enough types of resources for a proof of concept. We will construct a knowledge base for expanding goals into activities and demands on resources. We will implement sufficient plan surgery operations to enter, delete, and reexpand goals; to move activities to other resources; and to move activities in time. We will implement some plan surgery heuristics and tactical controls to control the plan surgery operations. The initial plan may be constructed not by the prototype but by some other planner such as SWITCH.

The user interface in the first prototype will include a color display representing activities on resource timelines. It will support approximately the same level of user editing as does Plan-It.

These one-year objectives describe the extent of our existing contractual obligations. However, based on a mutual agreement, we would propose in years two and three to perform the following work.

In the second year, we would attempt to discover and add more plan surgery heuristics from OR. We would improve the user interface to the heuristics, the interactivity between automatic planning, and plan editing by the user.

In the third year, we would attempt to strengthen the tactical control to allow the system to interleave OR heuristics and the AI heuristics.

If a follow-on effort as described above is desirable, we would submit a revised task plan and an updated cost estimate, as appropriate.

REFERENCES

[1] Vere, Steven, *Planning in Time: Windows and Durations for Activities and Goals* IEEE Transactions on Machine Intelligence, PAMI-5, No. 3, pp. 246-267, May, 1983.

[2] Webb, W. Allan, *Combinatorial Complexity of the Flight Project/Deep Space Network Resource Allocation/Scheduling Problem,* Operations Research Report, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California 1986.

[3] Glover, Fred, and Claude McMillan, *The Combinatoric Explosion: A Reassessment of Impacts of Microcomputers on Operations Research,* Elsevier, 1986.

[4] Minifie, J. Roberta and Robert Davis, *Survey of MRP Nervousness Issues,* Production and Inventory Management, Vol. 27, No. 3, 1986.

[5] Smith, Stephen, Mark Fox, and Peng Si Ow, *Constructing and Maintaining Detailed Production Plans: Investigations into the Development of Knowledge-Based Factory Scheduling Systems*, AI Magazine, Vol. 7, No. 4, 1986.

[6] Johnson, Craig, and David Werntz, *Automation of the Resource Allocation and Planning System at NASA's Jet Propulsion Laboratory,* presented at SPACE: Technology, Commerce and Communications, Houston, Texas, November, 1987.

[7] Meleton, Marcus P., Jr., *OPT-Fantasy or Breakthrough?* Production and Inventory Management, Vol. 27, No. 2, 1986.

[8] Biefeld, Eric, *PLAN-IT: Knowledge-Based Mission Sequencing,* Proceedings of SPIE on Space Station Automation, pp. 126-130, October, 1986.

[9] Fikes, R. E., and Nilsson, Nils, *STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving,* Artificial Intelligence Journal, Volume 2, pp. 189-208, 1971.

[10] Tate, Austin, *Review of Knowledge-Based Planning Techniques,* Knowledge Engineer's Review, Vol. 1, No. 2., British Computer Society, Specialist Group on Expert Systems, June, 1985.

[11] Sacerdoti, Earl, *Structure for Plans and Behavior,* New York, Elsevier North-Holland, Inc., 1977.

[12] Tate, Austin, *Project Planning Using a Hierarchical Non-linear Planner,* Department of Artificial Intelligence Report No. 25, Edinburgh University, 1976.

[13] Dean, Thomas, R. James Firby, and David Miller, *The FORBIN Paper,* Yale Technical Report 433, Yale University, 1987.

[14] Dean, Thomas, and Drew McDermott, *Temporal Data Base Management,* Artificial Intelligence Journal, Vol. 32, No. 1, pp. 1-55, 1987.

[15] Porta, Harry, *Dynamic Replanning,* Proceedings of ROBEXS '86, pp. 109-115, Instrument Society of America, 1986.

[16] Wilkins, David E., *Domain-Independent Planning: Representation and Plan Generation,* Artificial Intelligence, Vol. 22, pp. 269-301, April, 1984.

[17] Wilkins, David E., *Recovering from Execution Errors in SIPE,* Computational Intelligence, Vol. 1 No. 1, pp. 34-35, February, 1985.

[18] Fox, Mark S., B. P. Allen, and G. A. Strohm, *Job Shop Scheduling: An Investigation in Constraint-Directed Reasoning,* Proceedings of the Second National Conference on Artificial Intelligence, pp. 155-158, 1982.

[19] Porta, Harry, *SWITCH Users' Manual,* JPL Publication 86-24, Jet Propulsion Laboratory, California Institute of Technology, 1987.

[20] Stefik, Mark, *Planning with Constraints,* Artificial Intelligence Journal, Vol. 16, pp. 111-140, 1981.

[21] Stefik, Mark, *Planning and Meta-planning,* Artificial Intelligence Journal, Vol. 16, pp. 141-169, 1981.

[22] Minsky, Marvin, *Computation: Finite and Infinite Machines,* Prentice-Hall, 1967.

[23] Hayes-Roth, Barbara, *A Blackboard Architecture for Control,* Artificial Intelligence Journal, Vol. 26, pp. 251-321, 1985.

[24] Hayes-Roth, Barbara, Bruce Buchanan, Olivier Lichtarge, Mike Hewett, Russ Altman, James Brinkley, Craig Cornelius, Bruce Duncan, and Oleg Jardetzky, *PROTEAN: Deriving Protein Structures From Constraints,* Proceedings AAAI-86 Fifth National Conference on Artificial Intelligence, pp. 904-909, 1986.

[25] James, Mark, and Dave Atkinson, STAR*TOOL - An Environmental and Language for Expert System Implementation, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California, August 19, 1988.

ONE YEAR

JET PROPULSION LABORATORY

JPL

OMP Fiscal Year 1988
JPL Task Plan No. 81-2877
Prepared by H. Porta

	MILESTONES	4Q CY 87 / 1Q FY 88 Oct	Nov	Dec	1Q CY 88 / 2Q FY 88 Jan	Feb	Mar	2Q CY 88 / 2Q FY 88 Apr	May	Jun	3Q CY 88 / 4Q FY 88 Jul	Aug	Sep
1	Prepare research plan												
2	Develop functional requirements and												
3	produce f.r. document												
4	Develop preliminary evaluation												
5	criteria												
6	Problem formalization												
7	Identify state-of-the-art planning												
8	approaches												
9	Identify interface approaches												
10	Phase I design and preliminary												
11	implementation												
12	Phase I final implementation												
13	Review and update evaluation												
14	criteria												
15	Testing and evaluation												
16	Complexity study												
17	Prepare quarterly technical reports												
18	Prepare and present quarterly reviews												
19	Prepare final report												
20	Phase I demo												
21													
22													
23													

JET PROPULSION LABORATORY

OMP Calendar Years 1988-90

JPL Task Plan No. 81-2877
(for FY88 milestones see one-year milestone chart)
Prepared by H. Porta

	MILESTONES	Calendar Year 1988												Calendar Year 1989												Calendar Year 1990											
		J	F	M	A	M	J	J	A	S	O	N	D	J	F	M	A	M	J	J	A	S	O	N	D	J	F	M	A	M	J	J	A	S	O	N	D
1	Respecification for Phase II																																				
2	Prelim. Phase II design																																				
3	Redesign/update of Phase I																																				
4	Detailed Phase II design																																				
5	Phase II implementation																																				
6	Manual overide study																																				
7	Review & update evaluation criteria																																				
8	Testing & evaluation																																				
9	Respecification for Phase III																																				
10	Prelim. Phase III design																																				
11	Redesign/update of Phase II																																				
12	Detailed Phase III design																																				
13	Phase III implementation																																				
14	Review & update evaluation criteria																																				
15	Testing & evaluation																																				
16	Centralized/decentralized study																																				
17	Prepare quarterly technical reports																																				
18	Prepare & present quarterly reviews																																				
19	Prepare final reports																																				
20	End-of-phase demos																																				
21																																					
22																																					
23																																					

APPENDIX D

FUNCTIONAL REQUIREMENTS DOCUMENT

CONTENTS

1.0 INTRODUCTION

This document specifies requirements on the planning function of the Operations Mission Planner (OMP) for the Foreign Broadcast Information Service (FBIS). The Artificial Intelligence (AI) group of the Jet Propulsion Laboratory is conducting research on this resource allocation problem for the Central Intelligence Agency's Office of Research and Development (ORD). Theoretical design issues stemming from this document will be demonstrated through a series of prototypes. These prototypes will only partially implement the specified requirements.

1.1 SCOPE

The document is divided into eight sections. The Problem Definition, Section 2, enumerates the assumptions behind the FBIS resource allocation problem. In Section 3, the Philosophy or approach underlying the OMP prototype is outlined. This section discusses the theoretical attributes of the research approach. Section 4, Basic Planning, describes the planner operation flow by enumerating the input and output as well as the knowledge base representation characteristics. Planning Strategies, Section 5, then discusses the theoretical architecture which will operate upon the basic planning representations. Section 6, Interface, describes the interactive user interface. The implementation and run-time environments are outlined in Section 7, Hardware and Software Base. Finally, in Section 8, Acceptance Test Requirements, reference is made to a Preliminary Evaluation Criteria Study, which outlines the planner delivery standards.

1.2 CONVENTION AND NOTATION

Throughout the document, frequent reference is made to three concepts, System, Planner, and Plan. To clarify the ambiguity of these terms we define these concepts. The System is the world planning domain. This domain includes FBIS resources such as physical resources and personnel. The Planner is the software which schedules these resources. The primary output which it produces, the Plan, is a sequence of activities which optimizes fulfillment of a user-defined request.

1.3 APPLICABLE DOCUMENTS

Operations Mission Planner Research Plan; Porta, Biefeld, Falcone, 1987.
Operations Mission Planner Task Plan; Atkinson, 1987.
Space Flight Operations Center Sequence Subsystem (SEQ) Functional Requirements Document For Planning; Starbird, 1987.

1.4 THE ARTIFICIAL INTELLIGENCE PERSPECTIVE

The AI software development environment offers a new perspective to problem solving. Traditional planning techniques are not flexible. The AI emphasis is not upon a sequential progression from problem specification to implementation. On the contrary, an explorative approach is used in which problem expansion and implementation evolve together. Furthermore,

the AI process does not evaluate the complete solution spectrum. Instead, the process generates a manageable set of potential solutions from a host of concepts specific to the application.

2.0 PROBLEM DEFINITION

This section briefly describes the resource allocation problem incurred by the Foreign Broadcast Information Service (FBIS). It will describe the concepts and assumptions behind FBIS scheduling.

2.1 OVERVIEW

The FBIS scheduling environment offers a challenging spectrum of resource constraints and optimization strategies. Real scenarios are driven by anomalies. Pressing needs to monitor a hot topic require flexibility and timely interaction. In the operating environment a fast scheduler that replans with minimal disruption gives the strongest results.

This is the approach of the OMP prototype. When this scheduling problem is solved, a contribution can be made to a host of problems with various applications. No one has yet built a planner that replans with minimal disruption.

2.2 TENETS

The prototype design is focussed on meeting the following tenets:

1. Minimize lost collection.
2. Minimize perturbation of broadcast collection.
3. Perform continuous forward evaluation.
4. Plan with knowledge gained from previous decisions.
5. React quickly.

2.3 ASSUMPTIONS

In this section we describe our assumptions about the FBIS planning/scheduling/ resource-allocation problem. The planner is given knowledge about the resources (equipment and personnel) available, the desired tasks to be achieved (prioritized broadcasts to be collected), and the existing plan, if any. From these, the planner must, in a reasonable time, construct an executable plan to achieve some or all of the desired tasks. There are usually more tasks given than could actually be achieved with the available resources. The planner will turn out a plan that minimizes lost collection.

In section 2.3.1 we describe the types of tasks which the planner has to plan to accomplish. In section 2.3.2 we give a high-level description of the resources involved and their associated constraints. In section 2.3.3 we point out the interesting interaction between the prioritization of tasks and the imprecision of broadcast schedules and describe some strategies for dealing with them.

2.3.1 Task Description

The tasks whose achievement the OMP is to schedule consist of broadcasts to be collected and processed. The planner receives guidance with each task. The guidance includes specifications of the source (location and frequency), priority, and mode of processing of the broadcast. The guidance may also specify other aspects of the task, including time windows, required equipment, required personnel, and required level of fidelity. For some tasks, some of these aspects will not be present in the guidance, but will be available in a data base keyed by aspects that are in the guidance. For instance, the data base might contain an indication that all tasks to cover the speeches by Secretary Gorbachev have Radio Moscow as the source and are priority 2 tasks unless specified otherwise. The data base might link the title of a regularly scheduled broadcast to its start time, duration, and source. We will label this data base implicit guidance and, in this section, use the word guidance to include both this implicit guidance and the guidance that is explicitly provided in each individual task specification. (The sample tasks shown in this section are in English, while the first OMP prototypes will require that tasks be input in a more easily computer readable form. Some task examples given in this section are incomplete, leaving unspecified the aspects that are not directly illustrative of the principle under discussion.)

The OMP will also accept logical combinations of tasks of the above form. For example, "Relay the May Day Parades from Moscow and either Warsaw or Sofia."

The automated OMP will also accept tasks such as "Collect everything about the stock market in France" or "Monitor Iran." It will use a separate knowledge base to translate such tasks into sets of broadcasts to be collected.

In addition to allocating resources in order to fulfill the tasks of collecting specific broadcasts, the OMP must allocate some resources for search activities. The search operator scans the broadcast spectrum, looking for broadcasts that are not known as tasks but that are interesting nevertheless.

2.3.1.1 Priorities

The tasks to be scheduled have associated priorities. Each task's priority is a number from 1 to 100. Lower numbers indicate higher priorities. A task with a given priority number quasi-absolutely takes precedence over any tasks with greater priority numbers.

The possible exception to absolute precedence arises from the fact that priority numbers are assigned to entire broadcasts, while not all parts of a given broadcast are of equal importance. For instance, a situation may occur in which the FBIS is collecting a priority-six broadcast which will last another five minutes. Suddenly a new task arrives: to collect a priority-five broadcast beginning right away. However, the planner expects that the first five minutes of the new broadcast will consist of well-known background information. The planner may decide to collect the last five minutes of the priority-six broadcast instead of immediately switching to collect the beginning of the priority-five broadcast.

The OMP will require that tasks have priorities in the guidance. It will take the priorities into account when scheduling, giving high-priority tasks preference over low-priority ones in general, and making the exception described above when supplied with information about the relative importance of different parts of broadcasts.

2.3.1.2 Modes of Processing

When an antenna and receiver are receiving a broadcast signal, the received program can be processed in various ways. The program may be relayed to the United States, or a human monitor may listen to it and transcribe it or summarize it in writing. It may be recorded electronically for later relay, transcribing, summarizing, or archiving. The OMP will require that the guidance indicate the desired mode of processing for each task and will schedule the processing of each broadcast for which it schedules collection.

2.3.1.3 Time Windows

A task of collecting a broadcast should contain some indication of when the broadcast is expected to occur. This indication may be precise, in the form of an exact start time and (optional) end time or duration of the broadcast. However, sometimes these are not known exactly. For instance, maybe all that is known is that leader X will give a major speech in the next two days. In the worst case, maybe nothing is known about the time of occurrence of the broadcast except that it is some time in the future.

The OMP will accept a time window as part of a task's guidance. When no window is available from the guidance, the default window will be "some time in the future."

2.3.1.4 Level of Fidelity

There is a measure of level of fidelity, indicating the quality of a received signal, which is part of the guidance. The OMP's knowledge base must include sufficient facts about choosing resources to receive a broadcast with the desired level of fidelity. Given this information, the OMP will accept and respect the level of fidelity guidance.

2.3.1.5 Required Equipment and Personnel

Sometimes there are several choices of pieces or types of equipment and/or personnel that could be used to collect and process a broadcast with a sufficient level of fidelity. The originator of the task may not care which equipment or people are actually assigned to perform the task. On the other hand, the originator may list a choice of equipment and/or people as part of the task guidance. If so, it would mean that the task is not accomplished unless exactly that choice corresponds exactly to the original list of equipment and/or people. If the broadcast is collected and processed with other resources, that is not sufficient to fulfill this task, and it would be regarded as a waste of those other resources.

The OMP will accept and respect required equipment and personnel declarations in task guidance.

2.3.1.6 Other Guidance

At the discretion of the sponsor and the developers, the OMP will incorporate knowledge allowing it to deal with other guidance.

2.3.2 Resource Description

The resources which the OMP must allocate include antennas, receivers, and processors. The category of processors includes equipment, material, and personnel.

2.3.2.1 Antennas

The different antennas used by the FBIS have different limits on their ranges, both radial and directional. The plans produced by the OMP will respect these limits. The resource allocation heuristics will not schedule a broadcast to be received on an antenna when the broadcast's source is outside the antenna's range.

The different antennas also have different limits on frequency coverage. The plans produced by the OMP will respect these limits. The resource allocation heuristics will not schedule a broadcast to be received on an antenna when the antenna cannot receive the broadcast's frequency.

If an antenna is rotated sufficiently far in one direction, cable connected to it will wrap tightly around it and prevent it from rotating farther in that direction. The plans produced by the OMP will respect this limit.

A single antenna may be used to receive two or more simultaneous broadcasts from sources in the same, or nearly the same, direction. The plans produced by the OMP will take advantage of this capability. There will be resource allocation heuristics to find groups of simultaneous tasks that can be covered with and scheduled on a single antenna. (These heuristics will not always be active, depending on the urgency of producing a schedule quickly and other considerations.)

2.3.2.2 Receivers

The FBIS collects broadcasts on AM, FM, narrowband FM, facsimile, radio teletype, sideband, wire services, and television. It has receivers for each kind of signal. The plans produced by the OMP will show use of the right kind of receiver for each broadcast.

2.3.2.3 Processors

2.3.2.3.1 Processing Equipment

Processing equipment may include such items as tape recorders, blank tape, relay devices, typewriters, paper, dictionaries, etc. The OMP will have to schedule use of any resource

that is not regarded as automatically present wherever an activity is performed. For instance, it might be assumed that every person has a pencil or pen to use whenever he or she needs to write. In that case, the OMP does not need to have any representation of pencils and pens. On the other hand, it might not be assumed that every person has a reel of blank tape at hand at all times. In this case, when the OMP has to schedule an activity that requires use of blank tape, it must explicitly indicate that blank tape is needed, and how much. In addition, if it is anticipated that blank tape will ever be in short supply, the OMP will have to plan so that enough blank tape is available for the planned activities, i.e., so that it does not include more activities than the available amount of blank tape will support.

At the discretion of the sponsor and the developers, the OMP will include representation of various processing equipment resources. At a minimum, these resources will include tape recorders and relay equipment.

2.3.2.3.2 Personnel

People are involved in processing of received broadcasts in several ways. These may include monitoring, translating, transcribing, and summarizing the contents of broadcast programs. In addition, people are required to operate the equipment. Different people have different levels of skills with the different languages and subjects of foreign broadcasts and with the different kinds of equipment.

The OMP will plan so that appropriate personnel are assigned to each activity and no person is scheduled to work on too many activities at once or for more hours than allowed.

2.3.2.4 Other Entities Which OMP May Model as Resources

In addition to the above resources that are used directly to carry out activities, there may be other resources that the OMP will have to consider. As an illustration, if electronic equipment is powered from utility lines through a fuse box, the OMP will have to plan so that the equipment that is on at any one time is not drawing enough power to blow a fuse. As another illustration, if the equipment gets power from gasoline-powered generators, the OMP must plan so that enough gasoline is available to carry on the planned activities.

At the discretion of the sponsor and developers, the OMP will model all resources necessary to assure that plans are consistent and executable.

2.3.2.5 Connectivity

There are limits on the connectivity of resources from the various categories. For instance, not every antenna can be connected to every receiver. The OMP will include knowledge of the possible connections among antennas, receivers, and processors, and the capabilities of the various combinations for achieving the coverage, levels of fidelity, and processing called for in task guidance.

2.3.2.6 Variations in Resource Capabilities and Availabilities

Circumstances beyond the planner's control can change the capabilities or availability of resources. For instance, the availability or capability of a piece of equipment can be affected by breakdown or by its being pulled off-line for scheduled maintenance. The capability of a piece of equipment can be affected by meteorological factors or by jamming. The availability or capability of a person can be affected by illness or vacation. The OMP will accept information about such changes and plan accordingly.

2.3.3 Scheduling Trade-offs

The fact that there are almost always too many tasks for the existing resources leads to decisions to drop some tasks. The priorities given to tasks and the fact that a task might require collection of a broadcast whose time of occurrence is unknown can be conflicting influences on the choice of which tasks to drop. For instance, suppose that, on one morning at 7:30, Radio Moscow signs on with solemn music, an indication that some major official has died. A priority-three task is posted of finding out who died. It is presumed that Radio Moscow will eventually announce the name of the deceased, but that announcement could occur at any time of the morning. There is also a priority-ten task of listening to the daily propaganda announcement from Teheran at 9:00 a.m. The only receiver/antenna combination available that can receive Radio Moscow is also the only one available that can receive Teheran. Where should this equipment be directed at 9:00?

Different planning strategies would lead to different decisions. The strategy, "Let high-priority tasks have precedence over low-priority tasks," would cause the bureau to miss the Teheran propaganda broadcast, and there is the risk that the announcement from Radio Moscow would not come during that broadcast anyway. The strategy, "Prefer to collect broadcasts that are actually occurring over those that might occur," would cause the bureau to collect the Teheran broadcast at the risk of missing the announcement from Radio Moscow. Depending on the difficulty of switching between the configurations for receiving the two sources, there might also be a strategy, "Listen to the low-priority scheduled broadcast, but for the first ten seconds of every minute, switch to the source of the high-priority unscheduled broadcast. If the high-priority broadcast is there, stay tuned to it and collect it." As Radio Moscow would probably devote at least a minute to the obituary, and maybe ten seconds is long enough to determine whether a commentary is an obituary, this last strategy might be the most successful.

With experience and hindsight, a human planner might learn such relevant facts as the likely time for Radio Moscow to begin a major obituary (this might change from regime to regime). If the planner already had this experience, it might have enabled him to decide how closely to monitor Radio Moscow during the time of the Teheran broadcast. The automated OMP will incorporate such experiential knowledge where it is already available from experts. Moreover, the OMP will support modification of this experiential knowledge. Fully automatic modification of this knowledge would require a major advance in machine learning.

Other unclear decisions involve the collection of parts of broadcasts. For instance, if there are tasks to collect three broadcasts, an Iraqi body count from 8:30 to 9:00, a Polish farm

report from 8:30 to 8:45, and a speech by Secretary Gorbachev from 8:45 to 10:00, with priorities 8, 20, and 4, respectively, should the bureau collect the entire farm report or part of the body count at 8:30? There may be hard and fast rules, such as "It is always better to get a whole broadcast than a partial one, no matter what the priorities," which would apply in such a situation. Knowledge of the relative importance of parts of broadcasts (see section 2.3.1.1) could also be brought to bear.

3.0 PHILOSOPHY

The following section defines the design concepts behind the research approach. Replanning with minimal disruption is the research focus of the OMP prototype. This basic philosophy weaves several concepts together through implementation. Each concept carries a number of advantages.

3.1 REPLANNING

Adding changes to a refined plan drives the philosophy of replanning. Whenever a change in a task is desired, the new task is submitted to the planner. The planner generates a new plan. Conceptually this plan is a sequence of steps that accomplishes the desired task. Replanning, in this context, is the act of simply planning again.

In normal production, a schedule produced by the planner is an input into many other departments. These departments will use this plan to plan many other activities. The plan is also checked to see if it really does satisfy the requesters, and small changes may be requested to optimize some of the planning compromises. Many of these decisions lie outside of the planner's area of expertise.

In the course of operations many small changes to the original set of requested tasks will arise. If an entirely different plan is produced for each of these changes, much unnecessary work will be generated for the system as a whole. In fact, the amount of work generated may be prohibitively large.

When given a set of tasks, there exist many sequences which satisfy the requirements of the planner. If a slight change is made in the input, the planner will, in all probability, generate a very different schedule. This attribute of planning can render a totally automatic planner impractical.

A different approach to planning is to assume that planning is the act of modifying an existing schedule to reflect changes in the requested tasks. In this model of planning, the planner is working from an old schedule. The actions the planner takes try to update the plan in a minimally disruptive way. The planner can make small changes to the requested tasks without generating a whole new schedule.

Most planners approach replanning as simply planning again. There has been little work done on modifying old schedules to reflect changes in the requested tasks. Particularly hard

questions exist regarding how to control the search space in such a planner. This issue has had a major impact on the conceptual architecture of our approach to planning.

3.2 KNOWLEDGE BASE AND INFERENCE ENGINE

The domain in which a planner exists is never static. The types of tasks, the resources, and the rules of what is allowable are forever evolving. Thus, a planner needs to be easily adaptable. The common technique from artificial intelligence is to split the program into a knowledge base and an inference engine. The knowledge base contains rules on the domain-specific information. The inference engine is the code that manipulates the input using the rules from the knowledge base in order to generate the proper output. This technique has been used successfully in many different projects.

3.3 MULTIPLE PLANNING HEURISTICS

The concept of Multiple Planning Heuristics combines several planning techniques in one approach. Most planners use a single planning algorithm. The planning algorithm takes as input the requested tasks and any knowledge bases which describe what constitutes a legal plan. The planning algorithm then searches over the space of plans until it finds the appropriate plan. How the search is accomplished is entirely in the control of the single planning algorithm.

The planning search space of possible plans is extremely large. Any attempt to enumerate this space will consume extremely large amounts of time. Therefore, any planning algorithm must find the answer while controlling its search. There exists no known algorithm which can practically accomplish this control. The control of the search depends on the exact schedule the planner is working on.

The multiple planning heuristics approach tries to decompose the control problem into its own knowledge-base/inference-engine approach. Instead of using one algorithm, the different aspects of planning are controlled by different planning heuristics. This approach offers greater flexibility and sophistication in the control of planning.

3.4 ITERATIVE PLANNING

The control of planning depends on the schedule to be planned. The most powerful planning heuristics need to know approximately how the final schedule will be laid out in order to control the search. Thus, to plan, one needs to know how the plan will look. A successful technique is to first use a simpler approach to generate a schedule and then use this information to drive a second pass at the schedule.

This will also simplify the problem of how multiple heuristics can work on the same schedule. Each different heuristic can be viewed as another refinement of the schedule. The act of replanning using the original plan is another case of iterative planning.

3.5 USER-NATURAL INTERFACE

Expert scheduling personnel bring a host of strategies to a scheduling problem. Some scheduling software may schedule with routines and heuristics foreign to the personnel. The OMP prototype will not schedule in this manner. Instead, the prototype will allow the user to easily interact with the software because the software control will follow the same domain control heuristics as the expert scheduler. All representation and strategies developed in the design will reflect the user's natural perception of the problem.

4.0 BASIC PLANNING

The Basic Planning Model describes the inputs and outputs of the planner. The purpose of planning is to expand a set of Tasks into a Sequence of Steps. This sequence of steps forms a plan that can accomplish the requested tasks.

4.1 DEFINITIONS

4.1.1 Tasks

A single task corresponds to a goal that the system must accomplish. The accomplishment of a task will utilize some of the resources in the system. However, which resources and the exact time the resources will be used are not explicitly stated in a task. The purpose of the planner is to expand a set of tasks into specific steps.

4.1.2 Steps

A step corresponds to an action that must be taken in order to accomplish some task. A step has a definite start and stop time. It specifies which resources will be used and any other changes that must be made to the system.

4.1.3 Sequence

A sequence is a time-ordered series of steps. Each step in the sequence has a definite start and stop time. The execution of the sequence will accomplish the planned tasks.

4.1.4 Activities

It usually takes more than one step to achieve a single task. The series of steps that are used to achieve a single task is known as an activity. Thus, for each task in the plan there exists an activity which is composed of a few steps in the plan output.

4.1.5 Resources

The steps that accomplish a task consume various resources of the system. These resources can be physical devices such as receivers and antennas. They can also be more

abstract, such as the direction in which an antenna is pointed. The different activities must be scheduled so that these resources will not be overutilized.

4.1.6 Knowledge Bases and Inference Engines

The planner will be easily modifiable in order to support different and changing systems. To accomplish this, the software will be split into various Knowledge Bases and Inference Engines. A Knowledge Base describes the details of the system's domain to the software planner. This could include the resources and the different types of activities, among others. An Inference Engine will take an input and manipulate it by using the rules in the knowledge base which describe the domain. The decomposition of the planner into knowledge bases and inference engines allows the specifics of a domain to be an input to the program. Therefore, the planner's software would not have to be modified in order to adapt the program to a particular planning domain.

4.2 INPUTS

4.2.1 Old Sequence

The usual operation of the planner is to modify an existing plan. The planner will minimally disrupt the old sequence. If there is no previous plan, then this input would not be given. In this case, the planner conceptually starts with an old plan which is entirely devoid of tasks. To this empty set the planner will add the new tasks.

4.2.2 Old Tasks

The planner must be able to modify the old sequence in order to optimize the new tasks. In order for the planner to know how a step can be modified, the planner must know which task generated the step. Thus, the planner needs the original tasks from which the old sequence was generated. The planner will be able to link the old steps to the old tasks.

4.2.3 Changes in Tasks

This input directs the planner in what should be accomplished. This would include any new tasks, changes in existing tasks, and any task that is no longer desired.

4.3 KNOWLEDGE BASES

The knowledge bases support information that the planner needs in order to generate a legal plan. This knowledge does not, however, usually vary between different runs of the planner.

4.3.1 Task Expansion

The planner will generate a detailed activity for each requested task. This activity specifies a series of steps that can accomplish the requested task. This activity will be described to the planner by the Task Expansion Knowledge Base.

4.3.1.1 Activity Skeletons

The Task Expansion Knowledge Base will be composed of skeletons of various types of activities. A skeleton is an outline of an activity. A skeleton will include the number of, and relationships between, the various steps that compose the activity. For example, an activity skeleton could state that the activity would be composed of three steps, that the stop time of the first step is the start time of the second step, and that the three steps must use the same receiver. This activity skeleton would describe the resources used by each step and list all the options for each step. An option could be a set of devices from which the step could choose.

The inference engine associated with this knowledge base will find the activity skeleton that could be used for a given task. The inference engine will then form an activity from this skeleton. This new activity will be given the task's name and the various steps' parameters filled in from the set of legal options specified in the activity's skeleton.

4.3.1.2 Subactivities

An activity can be composed of both steps and subactivities. A subactivity is an activity that in turn can be composed of steps and subactivities. This forms a tree structure where the lowest leaves are steps.

The purpose of a subactivity is to collect a commonly used series of steps. Instead of repeating a common series of steps in several different activities, a subactivity skeleton could be described. A high-level activity could then just reference the subactivity.

4.3.1.3 Multiple Task Expansion

In most systems a requested task can be achieved by several different activities. The Task Expansion Knowledge Base may contain several different activity skeletons for each type of task. During the planning process the planner may try an alternative task expansion. The old task activity would then be replaced with a new task expansion. The new activity would keep as many of the old activity parameters as possible.

4.3.1.4 Subtasks

An activity can be composed of steps, subactivities, and subtasks. A subtask allows a high-level activity to have multiple expansion possibilities. This is convenient when a high-level task can be broken into a series of subtasks. Instead of developing several high-level activities, one high-level activity skeleton could be designed which uses several simpler subtasks. For example, a monitor task could be described by an activity that first sets up the equipment, then monitors the broadcast, and last of all tears down the equipment. There could exist five different setup activities and three different teardown activities. The high-level task could be described by

fifteen different high-level activities or by one high-level activity that contains a setup subtask and a teardown subtask. These subtasks will use the Task Expansion Knowledge Base in the identical way as the requested tasks.

4.3.1.5 Task Expansion Inference Engine

When given a task, the Task Expansion Inference Engine finds all the activity skeletons that can accomplish that task. The Task Expansion Inference Engine then picks one of the skeletons and expands this skeleton into an activity. This operation recourses through any subtasks in the chosen activity. The Task Expansion Inference Engine will also fill in any step parameters from the legal options so that a scheduled activity is formed. At any time the planner can direct the Task Expansion Inference Engine to reexpand any task or subtask under the direction of the planner.

4.3.2 Resource Knowledge Base

The resources available must be known by the planner. This would include all the antennas, receivers, and other resources that must be scheduled.

4.3.2.1 Resource Timelines

The resources will be represented by the planner as resource timelines. A resource timeline will exist for each resource in the system. These resource timelines will show how the resource usage varies over time. The resource timeline will also show any conflict in the resource due to oversubscription of the resource. The planner will be able to query the resource timelines for any steps which cause a resource conflict. This will allow the planner to deal explicitly with interactions between activities caused by oversubscription of the resources.

4.3.2.2 Resource Type Library

There are several different types of resources. For example, there are resources which are a pool of functionally identical devices. With this type of resource each step can ask for one or more devices. A conflict happens if, during any time interval, the number of devices requested exceeds the number available in the pool. Another type of resource models the consumption of some material, such as blank tapes. When a blank tape is used, it is not returned to the pool. A conflict occurs if the number of blank tapes used during the entire sequence exceeds the original number available at the beginning of the sequence. The planner will maintain a library of various types of resources. When an entry is added to the Resource Timelines Knowledge Base, it specifies a resource type described in the Resource Type Library. A new entry to the Resource Timelines Knowledge Base will contain the resource type, the resource name, the amount of resource available, and any other parameters appropriate for the resource type.

4.3.2.3 Resource Inference Engine

The Resource Inference Engine calculates the resource usage through time by monitoring all the steps that use any particular resource. Whenever a step is added, modified, or deleted, the

inference engine quickly updates all of the resource timelines that are affected. The inference engine will check for any changes this causes in the conflicts of the affected resources. Another duty of this inference engine is to maintain the necessary statistics on the resource usage and conflicts. This information will be used to identify the usage or conflict patterns in the resource timelines.

4.4 OUTPUT

The main output of the planner is a detailed timeline. To strengthen planning productivity, the planner will also produce summary reports. To support replanning, the planner will generate files which describe the sequence links and the history of conflict resolution.

4.4.1 Timeline

A timeline will display the resource assignments by time. Activities will be represented over time and synthesized over an orthogonal column of resources.

4.4.2 Summary Reports

Summary Reports will compile operation statistics and parameters to describe the prototype efficiency and the schedule development history. Reports are distinguished by their application domain. For example, there are reports used by FBIS personnel, such as Performance Reports, and reports used for replanning, such as History Reports.

4.4.2.1 Performance Report Generation

Various hard-copy representations of the schedule performance data will be provided via Report Generation menu options.

4.4.2.1.1 Resource Use Profile

This profile will describe the resource allocation by using histograms. Each bar of the histogram will represent a resource and will summarize the total usage of the resource and the absence or presence of a resource conflict.

4.4.2.1.2 Activity Profile

Indexed by activity, this profile will enumerate the resources used to conduct a user-specified activity.

4.4.2.1.3 Step Profile

A step profile will generate a list of one-line descriptions to describe each step necessary to implement a task. The user will specify the task name.

4.4.2.1.4 Gap Report

The Gap Report identifies all unused periods of time longer than any user-specified period. This helps to find periods where activities can be added to a crowded timeline and in finding patterns in link placement which may help identify strategies for more efficient utilization of time.

4.4.2.2 History Reports

History Reports will be used to replan future sequences. Therefore, they will be generated in machine-readable format and fed into the planner at the beginning of the next replanning session. These reports will document the task expansion, activity expansion, and conflict resolution progression.

4.4.2.2.1 Task Expansion Report

To help the planner modify the generated timeline and optimize new tasks, this report will describe task and activity expansion. The data will be organized with a tree-structure format. The output format will remain in machine-readable structure for replanning purposes.

4.4.2.2.2 Sequence

To represent the timeline for future modification, the planner will generate a timeline sequence in machine-readable format. The sequence will use a tree-structure format to represent activities and resource allocation.

5.0 PLANNING STRATEGIES

The above Basic Planner can represent plans. The basic planner contains the knowledge of the requested tasks, the different activities that can satisfy a task and how the activities compete for the system resources. What is missing is how the planner can use this information to create a workable schedule. Planning Techniques, Plan Chronology, Planning Focus, and Resource Pattern Recognition describe how to develop the plan representation into a working schedule.

5.1 PLANNING TECHNIQUES

Instead of having just one algorithm to assemble the plan, the planner will be composed of a large set of planning techniques. The planner will invoke those planning techniques which will manipulate the candidate sequence by using the basic planning representations.

5.1.1 Planning Algorithms

Planning Algorithms are large algorithms which totally solve some aspect of the plan. These are typified by operations research programs such as dynamic programming or simulated

annealing. These planning algorithms try to completely solve some part of the plan. They completely control their own search of the plan.

5.1.2 Planning Heuristics

A planning heuristic is concerned with only one small aspect of the plan. There exist different heuristics to control the expansion of a task, versus heuristics controlling which resources an activity will use. There can be several layers of heuristics working on a single parameter, such as the start time of an activity.

For example:

5.1.2.1 Coarse Step Positioning

This type of heuristic would place an activity in a time region on the timelines. The heuristics' objective is to schedule a task where there is not much competition for the task's required resources.

5.1.2.2 Fine Step Positioning

This type of heuristic places a step with respect to the local resource usage. These heuristics would be triggered, depending on the pattern of the local resource usage.

5.1.2.3 Micro Step Positioning

It is desirable to exactly align a step to the natural borders in the sequence. This will be accomplished by a set of heuristics that would either place the step at the edge of the step's timing requirements or place the step on the edge of another step which uses the same resources.

When the human user is graphically editing a schedule, the mouse (see section 6.2.1.1.2) is useful for selecting the step to move. However, for placing the step at an exact point in time, the mouse is not accurate enough. The micropositioning heuristics could relieve the user from having to exactly specify the time. They would automatically cause the step to snap to the exact position.

5.2 TRIGGERING PLANNING TECHNIQUES

The act of planning involves invoking the proper planning techniques in the proper order. The planning techniques use the basic planning representations to manipulate the sequence. The triggering of the techniques depends on the particular candidate sequence and the state of the planner.

5.2.1 Triggering by Desirability

It is more desirable to reassign the resource a task uses than to delete the task. However, deleting the task does free the resource for other usage. These different heuristics trade off this desirability for power and ease in planning.

5.2.2 Triggering by Resource Usage

If a type of resource is vastly overused, then swapping steps within the resource will not be very productive. The planner should delete tasks until the resource usage becomes more reasonable. The planner will use the various patterns in resource usage to trigger planning techniques.

5.2.3 Triggering by the Amount of Effort

Given a choice, the planner will choose to work on tasks on which the least amount of effort has been expended.

5.2.4 Triggering by Previous Scheduling History

If a step has just been moved, a microadjustment may be in order. If a scheduling strategy has been pursued for some time without noticeable improvement, then it is time to try something else.

5.2.5 Triggering by Focus Decisions

The planner may decide to focus on conflicts of some set of resources before checking the other resources. It would trigger appropriate heuristics to work on the critical time region. These focusing decisions are made by a special group of metaheuristics that control the planning process.

5.3 CHRONOLOGY

The planner starts with an old plan and some task changes to add to the schedule. The planner then goes through several stages in modifying the schedule. The planner must track these different stages. This tracking of the various stages is the chronology of the schedule.

5.3.1 Definition of Time versus Chronology

Time is a parameter of steps and activities. Chronology is tracking the stages that the planner goes through while modifying the plan.

5.3.2 Chronogram

A chronogram represents a stage of the planning process. The chronology is composed of a series of chronograms.

5.3.3 Levels of Chronology

The planner will track these planning stages on several levels. The top level shows how the plan as a whole is progressing. When the planner focuses on a particular time region, the work done on this time region will be represented at a lower level in the chronology. Within this time region, the planner may progress through several planning strategies. Each planning strategy is a different chronogram on a lower planning level. The cycles within a strategy would represent an even lower level.

5.3.3.1 Top-Level Chronology

The top-level chronology tracks the major stages of the entire schedule. Near the beginning of this level, the planner will allow large-scale changes to the schedule but will not allow much effort to be expended on any one task. As the top level evolves, the planner will restrict the scale of any changes made to the schedule but will allow a lot of effort to be expended on a few tasks.

5.3.3.2 Medium-Level Chronology

The planner will track how long a planning strategy has executed. After a set period, a chronogram will be posted that will trigger heuristics, which will determine if the current strategy is making progress or if another planning strategy should be pursued.

5.3.3.3 Low-Level Chronology

While the planner is working on a single activity, there could be several different layers of heuristics which refine the activity's parameters. These layers of heuristics could represent the different accuracies of adjusting an activity's start time. For example, it is necessary to adjust the micro time position of a step only after the activity has been localized to a new time region.

5.3.4 Task Chronology

The chronology of a task tracks the effort that has gone into scheduling that task. This information is used by various planning heuristics to determine which task to work on next. The major changes in a task will also be tracked. This is to keep the planner from retrying old alternatives. However, as the planner is increasingly willing to invest a greater effort on a task, the planning heuristics could retry old alternatives.

5.3.5 Resource Chronology

A resource's chronology tracks the effort that has gone into reducing conflicts on a particular resource. This chronology will be tracked not only for the resource as a whole but also for interesting time regions of the resource timeline. This allows the planner to look for patterns of change in a time region of the plan. For example, if the conflict level within a single time

region keeps cycling up and down, the planner may decide to focus on that region and then freeze the results.

5.4 PLANNING FOCUS

The planning focus is the mechanism of communicating the metaheuristics' decisions to the planning heuristics. The metaheuristics may decide to reduce conflict by deleting tasks from a certain region of the sequence. By posting the proper focus decisions, the appropriate planning heuristics will be triggered.

5.4.1 Focus and Chronology

The chronology tracks the stages of the planning process, while the focus tells the planning heuristics what to work on next.

5.4.2 Focus Levels

The focus levels parallel the levels of the chronology. This is to allow different sets of heuristics to control the various tactical levels of the planner.

5.4.3 Focus Expectations

When the focus is set by some metaheuristics, the expectation for the focus is recorded. The expectation states under what conditions the focus should be abandoned. One of these conditions will always be triggered by the chronology. This is to prevent the planner from working too long on any one strategy.

5.5 METAHEURISTICS

The metaplanning heuristics implement the strategies of the planner. They accomplish this by controlling the focus state of the planner.

5.5.1 Levels of Metaheuristics

The different focus levels would be controlled by different metaheuristics. Some of the metaheuristics within a level would start a focus level, while others would terminate the focus. The metaheuristics of a lower focus level would be controlled by the focus set at higher levels.

5.5.2 Triggering of Metaheuristics

The metaheuristics are triggered by specified changes in the chronology of the planner. They then use the patterns in the resources, the chronology of the planner and the chronology of the tasks to determine the next focus state.

5.6 RESOURCE PATTERNS

Much of planning is dependent on the patterns of the resource usage. Instead of directly checking for the patterns, the heuristics will use special-purpose routines that can classify the resource timelines. Using these routines, the planner would automatically maintain a symbolic description of the resources.

5.6.1 Global Patterns

These include the total resource usage of a resource timeline and a measure of the resource usage deviation. The deviation is a useful measure for resource leveling.

5.6.2 Local Resource Usage

Focusing on local resource usage allows the planner to avoid areas of high resource usage and to select areas where the resources are readily available.

5.6.3 Special Patterns

These pattern routines will find special patterns in the resource usage. An example of a pattern is a hill in the resource usage which peaks over the conflict level. Another such pattern example is a sawtooth usage pattern. Different planning tactics would be triggered to handle these different patterns.

5.6.4 Chronology Patterns

As the plan evolves, the resource usage in a particular time region may cycle. This may not be relevant at the beginning stages of planning. However, in latter stages these cyclic regions become interesting to the planner and will trigger specialized strategies.

5.7 STRATEGY POSTMORTEM

If a focus strategy does not accomplish its objectives, the planner does a Strategy Postmortem. The postmortem helps the planner to decide what would be a useful focus to try in order to achieve the higher objectives.

5.7.1 Why Strategy Failed

The first part of the postmortem records why the current focus was abandoned. This could be because it did not make enough progress for the effort expended. Alternatively, the higher level focus may have been too restrictive, or maybe the heuristics could not find any alternatives to try.

5.7.2 Recommendations

The second part of the postmortem includes recommendations on what to try next. This could be to loosen up on the focus level to allow more possibilities. However, the postmortem will only suggest to loosen up on the focus parameters that failed the previous heuristics. Another recommendation would be to try again but to allow more effort to be expended, or to try a set of more powerful heuristics. Finally, the Strategy Postmortem could tell the planner to simply give up on this set of higher level objectives.

6.0 INTERFACE

The planner will support interactive editing of candidate sequences. To support this, the planner will provide graphical displays of the sequence and the resources. The planner will allow the user to graphically or textually edit any step, activity, or task and will immediately update the appropriate displays.

6.1 BACKGROUND

Much of scheduling is spent in making compromises between competing tasks. Most of the time of the user is spent in deciding these compromises. This process may well involve several of the individuals involved with the competing tasks. The compromises may be political in nature and reflect knowledge well beyond the scope of the planner. The planner shall support this type of process.

6.2 DISPLAYS

The planner will display the current candidate sequence to the user.

6.2.1 Timeline Displays

The planner will be able to present the candidate sequence in a graphical timeline format. There will be many different types of timeline displays, each of which specializes in presenting a different aspect of the candidate sequence. These graphical timeline displays will be able to plot either activities or resources.

6.2.1.1 Zooming and Panning

The user will be able to zoom the graphic timeline displays to various magnifications and to pan through the timeline displays to different time intervals.

6.2.1.1.1 Quantized Zooming and Panning

The timeline displays will be able to present varying time intervals of the sequence. One such time interval will be the whole duration of the candidate sequence. The user will be able to zoom into a portion of this duration. Then the user will be able to pan the timeline displays into other time intervals. The planner will support discrete zooming and panning of the graphic

timeline displays. With the use of a master menu, the user will set the different duration intervals to which the timeline displays can be magnified. The user will also set a panning duration which is used to quantize the panning of the displays. The quantized duration interval of a pan shall be specified for each of the zoom magnifications. For example, a candidate sequence may have a total duration of one week. The first zoom magnification is therefore one week, and there is no panning duration for this magnification. The second zoom magnification may be one day with a pan duration of 12 hours. At this magnification the timeline displays will plot only a 24-hour duration, and the displays will always start either at 0:00 or at 12:00 of any day in the candidate sequence. The final magnification of zoom may be four hours with a pan duration of one hour. At this magnification the timeline displays will show 4-hour intervals of the candidate sequence with the start time on any hour boundary within the sequence.

6.2.1.1.2 Panning Display

To support panning and zooming of the timeline displays, there will be a temporary pop-up display which will graphically present the current display interval with respect to the rest of the candidate sequence duration. The user will be able to graphically pan or zoom the timeline displays with the use of this display. The display will also plot a time scale for use in graphically determining the temporal position of a plotted timeline item. The control of this display will be accomplished with a graphic pointing device (mouse).

6.2.1.1.3 Timeline Synchronization

The graphic timeline displays will normally be synchronized. This means that the timeline displays will simultaneously present the same time interval. However, the user may be able to freeze any display to a particular time interval. When panning or zooming the rest of the displays, the frozen display will not change its displayed time interval. It will be possible to explicitly pan or zoom a frozen display. When the display is unfrozen, it will be synchronized to the rest of the timeline displays.

6.2.1.1.4 Magnification-Dependent Plotting

It will be possible for the different timeline displays to use different plotting functions at the different zoom magnifications. This will allow the displays to show more detailed information at the higher magnifications of the timelines. For example, see section 6.2.1.2.2.

6.2.1.2 Activity Displays

There will be timeline displays of the activities in sequence.

6.2.1.2.1 Activity Plotting

Activities will be presented as labeled line segments on the timeline display. The line segment end points will correspond to the start and end times of the activity's first and last steps. The activity's name will be used to label the activity line segment.

6.2.1.2.2 Step or Subactivity Plotting

At higher magnification, it may be desirable to show an activity's steps or subactivities. The activity will then be plotted as a collection of steps or subactivities. Each step or subactivity will be plotted as specified in 6.2.1.2.1. Whether the activity is displayed as a single line segment or as a collection of line segments will depend on the current zoom magnification.

6.2.1.2.3 Activity Display Types

The planner will support an activity timeline display for each activity type and for various collections of activity types. Thus, all the activities need not be in any one display, but the planner will divide the activities between several displays, depending on the activity type. Each activity display will contain a description of the presented activity types. This description will specify the exact configuration of the plot at each zoom magnification.

6.2.1.2.4 Selecting Activities

The user may point to an activity using the mouse. The user may then select the activity for editing or for graphically changing the activity location in the candidate sequence.

6.2.1.3 Resource Timeline Displays

There will be a timeline display for each resource in the planner.

6.2.1.3.1 Resource Plotting by Type

The plotting function used for a resource will depend on the resource type. Some resource types may be plotted as histograms, while others may be plotted using color coding or pixel patterns. It will be possible to have several different plotting functions for the same resource type. In this case, there will be a different resource timeline display for each plotting function. The resource plotting functions will be part of the resource type library (section 4.3.2.2).

6.2.1.3.2 Resource Conflicts

The resource timelines will prominently mark new conflicts. The planner will contain special panning commands to automatically pan the timeline displays to the resource conflict intervals.

6.2.1.3.3 Querying Resource Timelines

The user may use the mouse to query the resource timelines. The resource timelines will support queries for the numeric and symbolic amount of resource usage. Selecting steps or activities at a particular point in the resource timeline will be supported.

6.2.2 Edit Displays

The planner will support various types of displays to allow the user to modify the sequence and control the action of the planner. The user will be able to call up these displays by selecting the sequence item to be edited. The display will normally disappear after the user has finished editing this item. However, the user may force the display to be a permanent part of the display layout (section 6.2.4).

6.2.2.1 Step Edit Displays

The planner will include displays that present the parameters of a step. These will include the step's start time, stop time, resource usage, and any other parameters. These displays will allow the user to textually modify any of these parameters.

6.2.2.2 Activity Edit Displays

These displays will present the internal parameters of an activity. They will allow the user to change any of the activity's parameters or to select one of the activity's steps for further editing. These edit displays will be presented in long and short formats. The long format presents all the activity's parameters, while the short format presents only the most commonly modified parameters.

6.2.2.3 Task Edit Displays

The task edit displays allow the user to examine and control the task expansion of a requested task. These task displays can also be used to add, delete, or modify a task.

6.2.3 Planning Control Displays

The planner will support displays to present to the user the processing state of the planner including displays of the chronology of the plan and the current focus states. The user will be able to modify the current focus states in order to direct the actions of the planner.

6.2.4 Display Layouts

The layout of the displays, described in section 6.2, will be under the control of the user. The user may call up, remove, or position any of the displays. The user may save any layout. A saved layout describes the display types, their positions on the screen, and the sequence items that they display. When the user loads a saved layout, the planner will create the appropriate displays, position them on the screen, and update the contents.

6.3 COMMANDING

The planner will support several different command modes.

6.3.1 Menus

Commonly used commands will be executable by menu items. The mouse will be used to invoke the menus. Menus of commands about activities will be accessible by selecting the activity from either an activity timeline display or a resource timeline display.

6.3.2 Typed Commands

Every command, including those usually executed by the mouse or menus, will have a typed form. Whenever the command is executed, the planner will present the typed form of the command in the command line display. The user will be able to build files containing lists of these commands, which could later be executed.

6.3.3 Command Parameters

There will be several methods in which a command parameter could be specified. These methods will include typing, selecting from a display with the mouse, and menu items. Any of these methods can be mixed during the entering of a single command. For example, a command to move an activity could be selected from a menu. The activity to be moved could then be selected from a resource timeline display with the mouse. The point where the activity is to be moved could be typed into the command line display. Before the command is executed, the command line display will present the typed version of the command, including all the appropriate parameters.

6.4 AUTOMATIC PLANNING SUPPORT OF MANUAL EDITING

The user may enable any part of the automatic planning mechanism as desired. This could include the low-level focus techniques. The planner would then make small adjustments to any editing the user makes. This would be used to relieve the user from exactly specifying an activity's step end times. The user could then use the mouse to position an activity, and the automatic planner would finely adjust the results as specified in section 5.1.2.3.

7.0 HARDWARE AND SOFTWARE BASE

7.1 IMPLEMENTATION ENVIRONMENT

Development of the OMP prototype will be conducted on Symbolics or Texas Instruments machines with Common LISP software and a selection of software tools. The tools include STAR*TOOL, Plan-It, SWITCH/Runaround, Time Map Manager, FORBIN and BB1. For comprehensive tool descriptions, see Appendix A, the Operations Mission Planner Research Plan.

7.2 RUN-TIME ENVIRONMENT

The prototype will run on the Symbolics or Texas Instruments machine under Common LISP software.

8.0 ACCEPTANCE TEST REQUIREMENTS

The acceptance test requirements are enumerated in the study entitled Operations Mission Planner Preliminary Evaluation Criteria.

APPENDIX E

OMP I vs OMP II
COMPARISON OF PROTOTYPE
CAPABILITIES

Attribute	OMP I	OMP II
Number of Tasks	75	500–1000
Types of Resources		
Antennas	3 types	4 types
Receivers	0	0
Relays	0	1 type
Recorders	0	0
Translators	0	3 types
Resource Representation		
Capacity	no	yes
Direction/State	yes	no
Consumables/Replenishables	n/a for FBIS	n/a for FBIS
Iterative Refinement Phases		
Initial Load	yes	yes
Resource-Centered	yes	yes
Bottleneck-Centered	no	yes
Optimization	no	yes-partial
Event Handling	no	yes-partial
Interleaving of Phases	no	yes
Control Levels		
Executive/Master	yes-partial	yes-partial
Strategic	no	yes
Tactical	no	yes
Heuristics		
Assessment	no	yes
Dispatch	yes	yes
Control	no	yes
Bottlenecks		
Identification	yes	yes
Classification	no	yes
Types of Tasks		
Basic	yes	yes
Alerts	yes-partial	yes
Resource Events	no	yes

Attribute	OMP I	OMP II
Task Representations		
Time Windows	single, fixed	multiple, variable
Steps	yes	yes
Activity Structure	no	yes
Scheduling Processes		
Scheduling Engine	yes-partial	yes
Search Engine	no	yes
Chronologies	yes	yes
Scheduling Actions		
Move	yes	yes
Delete	yes	yes
Undelete	no	yes
Gapping	no	yes
Shrinking	no	yes
Hand-Offs	no	yes
Panning	yes	no
Interface Features		
Timelines	yes	yes
Histograms	yes	yes
Strip Charts	no	yes
Directions	yes	no
Priority Coding	yes	yes
Time Cursor	no	yes
Messages	no	yes
Phase Display	yes	yes
Strategy Display	no	yes
Edit Window	yes	yes
Activity Graph Window	no	yes
System Commands	yes	yes
Mouse Sensitivity	yes	yes

APPENDIX F

STATE OF THE ART IN AI PLANNING

1.0 INTRODUCTION

This paper addresses the state of the art in Artificial Intelligence (AI) based planning. The discussion covers a range of activities in the area of planning research but focuses on those topics which are applicable to the Operation Mission Planner (OMP). The major emphasis is on planning and scheduling techniques developed for job shop scheduling, although several of the systems discussed have emerged from other planning domains, such as robot control, experiment generation, design, and production planning.

The current state of the art in planning has evolved through a series of advances in both the "plan representation" and the "planning process." A plan can be viewed as a set of ordered steps which accomplishes the specified goals. The planning process is the series of decisions that the planner makes in generating the plan. As the state of the art becomes more advanced, understanding and capitalizing on the differences between the plan and the planning process become more sophisticated.

The paper is divided into two major sections: classical and alternative approaches to planning. For our purposes, classical planners are general purpose planners and are based on goal decomposition. Alternative planners are specialized systems which have marked distinctions between the plan and the planning process representations. Because of the evolutionary growth in AI-based planning, the boundaries between classical and alternative planners are often fuzzy.

2.0 CLASSICAL PLANNERS

Planning has evolved as a special case of general AI-based problem solving. The early classical planners based their model of planning upon the General Problem Solver (GPS) [15]. The GPS approach to problem solving is to decompose a problem down into smaller, more easily solved subproblems. Classical planners use this concept in goal expansion. If the planner cannot immediately satisfy a goal, it looks for a rule that can achieve the goal. The rule will have a set of preconditions that must be true in order to execute the rule. Each of these preconditions becomes a new subgoal for the planner. The goals and subgoals are hierarchically linked, where each goal is decomposed into its descendants. After all the original goals and subgoals are satisfied, the rules that were used in the decomposition of the goals are translated into the plan steps.

This hierarchical tree of goals and subgoals represents the plan. The process of building the tree and deciding which rule to use to expand a goal is the planning process. The current state of the planning process is directly reflected by the current state of the plan tree. In this type of planner the current plan and the planning process are tightly coupled.

The set of all possible plan trees is called the solution space. The solution space contains the final plan along with all the possible intermediate states of the plan. The planning process searches this conceptual space for the final answer. This concept of searching a space of possible solutions is common to all planners.

The basic problem common to all planners is that the solution space grows exponentially with the number of rules and goals. This leads to searches that take extremely large amounts of time. One major thrust in planning research is how to accomplish this search in a reasonable time span.

One of the earliest planners to use the concept of goal expansion is STRIPS [10]. In order to simplify the problem, STRIPS assumes that goals are linear. In other words, in order to achieve two preconditions A and B, the process must first achieve subgoal A then achieve subgoal B. This assumes that the steps that accomplish the goals do not interact in such a manner that the subgoals A and B may have to be ordered as first B then A.

STRIPS starts by selecting a rule with which to expand the first goal. STRIPS then expands the first subgoal created by the first rule. It continues expanding subgoals until the first goal is completely solved. STRIPS then proceeds to the second goal. This is a depth-first search of the solution space. During this search the planner can reach a point where it cannot find a viable rule. The planner then undoes its last goal expansion and tries a different rule. In this manner STRIPS can explore the entire solution space until it finds a viable plan. This type of exploration is referred to as chronological backtracking because the decisions are undone in the reverse order in which they were made. If an incorrect decision is made but not discovered until much later in the planning process, the planner is forced to search through a very large portion of the solution space.

In an attempt to alleviate some of the problems associated with chronological backtracking, the concept of planning by using a hierarchy of abstraction spaces was introduced in an extension of STRIPS called ABSTRIPS [31]. In planning systems, hierarchy refers to two related concepts. Most planners have a hierarchical structure which describes the relationship of goals and subgoals. This is not to be confused with hierarchical planners which use a hierarchy of abstraction spaces in developing a plan. ABSTRIPS concentrates on forming a solution at the most abstract level and then later fills in the lower level details. Thus, it does not expand a top level goal all the way down to its lowest level's subgoals. Instead, after reaching the appropriate depth for the current abstraction level it proceeds to the next top level goal. After completing a plan at this level of abstraction it proceeds to the next level of goal abstraction and expands the subgoals generated by the previous level's search. This concept allows the planner to first work out a high-level abstract plan before working on the exact details of the plan. Certain preconditions can have higher priorities associated with them. This allows ABSTRIPS to focus its attention on the most important subgoals first.

Linear planners such as STRIPS and ABSTRIPS do not perform well with real-world problems because interactions generally exist between goals [4]. The interactions between different goals must be recognized and addressed. Many of the interactions can be resolved by proper ordering of the plan steps. Instead of immediately specifying the order of the plan steps, nonlinear planners, such as NOAH [32], keep the steps in a parallel network until forced to specify an ordering.

The delaying of the ordering of plan steps is a type of least commitment planning. NOAH uses least commitment to reduce the amount of backtracking needed to find a valid plan.

By avoiding commitments early in the planning process the planner can wait until planning constraints force a decision. Otherwise the planner would have to backtrack to the point where the decision was made.

NOAH also incorporated the use of planning critics into its planning process. These critics analyze a plan and try to identify and correct goal interactions without backtracking. The critics can directly manipulate the plan network. Some of the critics check for nonlocal goal interaction and specify the ordering of the planned steps. Critics are also used to place domain-specific knowledge into the planning process. The critics exist outside of the plan and are only a part of the planning process. This allows the hierarchical planner to construct a cheap, but incorrect, plan by just considering local detail and then using the critics to check for inconsistencies in the plan caused by nonlocal goal interactions.

Another approach to containing the search is through the use of plan skeletons. MOLGEN [11], a planner used in constructing molecular genetics experiments, uses plan skeletons to simplify the search space. In most planning domains it takes several rules to accomplish a single top-level goal where each rule represents a different step in the plan. Each time the planner tries to satisfy a goal it must rederive the proper set of steps from its set of rules. However, there may exist only three or four different combinations of steps that can actually accomplish the goal. A plan skeleton specifies the series of steps that accomplish the stated goal. This skeleton specifies which parameters need to be filled in and any constraints between the parameters. Thus, the planner is saved from inventing the plan segment from scratch.

Because chronological backtracking evaluates all alternative decisions, it must rediscover contradictions and regenerate successful paths. A more efficient approach to the search is dependency-directed backtracking [12]. In chronological backtracking, the search engine and the fact database are usually totally integrated. In dependency-directed backtracking the implementation of the fact database is a separate program known as the truth maintenance (or belief revision) system.

Whenever a fact is asserted, the truth maintenance system records which other facts this new fact depends upon. The truth maintenance system then determines if the new fact is consistent with the rest of the fact database. If an inconsistency arises, the truth maintenance system informs the search engine. The search engine can query the truth maintenance system to find the inconsistent facts and the facts upon which they depend. The search engine can retract any of these facts to eliminate the inconsistency. The truth maintenance system will eliminate only the facts which depend on the retracted fact. If a retracted fact is ever reasserted, the truth maintenance system will automatically reassert any previously derived facts that depended on the reasserted fact. By using a truth maintenance system a search engine can implement dependency-directed backtracking. Instead of undoing decisions in the order in which they were made, the dependency backtracking search engine will only undo decisions that are relevant to the discovered inconsistency. Also, the system will remember previous successful paths instead of rederiving them. The major function of the truth maintenance system is to ensure that a program is reasoning with consistent information.

Earlier planners viewed time as a simple ordering of the steps in a plan. These planners can specify that step A has to happen before step B but cannot specify any exact times or durations. DEVISER [41] introduced a simple concept of absolute time to planning systems. DEVISER can represent the fact that a step has to start between 3:00 and 5:00 and that the step will take 30 minutes to complete. As DEVISER orders the steps, it reduces the possible start windows of the steps until it has assigned an exact start time for each step in the plan. DEVISER does not, however, have a completely flexible representation of time. It cannot, for instance, reason effectively about a plan step whose duration is a function of its exact start time.

Temporal calculus serves as a theoretical baseline for development of advanced time-reasoning systems [29]. Temporal calculus is a mathematical formalization which enables reasoning about time and the partial ordering of plan steps. The planner can assign absolute or relative times to the planned steps and check for consistency in the assignments. For example, the planner could assert that Step A must be separated by 5 minutes from Step B. If Step B starts at 5:00 and lasts for one hour, then Step A must either end at 4:55 or start at 6:05. In theory this automatic inferring of all possible relationships between the steps is an NP-complete problem [9]. However, some systems, such as the Time Map Manager (TMM) [7], are polynomial rather than NP-complete in their search. This means that the time that it takes to answer a query is a polynomial function of the number of facts. Thus, the TMM cannot determine all possible relationships but can find most of them in a reasonably efficient manner.

FORBIN [8] is a planner built around the TMM; FORBIN incorporates many classical planning techniques. The TMM serves as a temporal truth maintenance database for FORBIN. FORBIN uses a plan library of plan skeletons. When a new goal is selected, a skeleton is found in the plan library. The goal expander expands the goal by filling in the parameters in the plan skeleton. This expansion is done in a hierarchical fashion which allows the TMM to represent the whole future to some level of refinement. The TMM uses the constraints to make temporal predictions on plan viability. However, the TMM is not capable of detecting all of the steps' interactions. Therefore, FORBIN has a scheduling module which searches the partially constructed plan for a totally specified schedule. If a viable schedule is found, FORBIN goes on to the next goal to be expanded. Thus, the scheduler module in FORBIN acts as a checkout procedure to ensure that the plan constraints generated so far have at least one possible solution. The particular parameters chosen by the scheduler will not be enforced, in keeping with a policy of least commitment.

The evolution of classical planners has been plagued by the search times associated with remaining general purpose in nature. While several systems have made substantial progress in this area, they are still unable to solve large real-world problems. The problem is what mathematicians refer to as NP-complete, which means that any solution in general must use an exponential search [42]. Most researchers agree that any practical solution must use large amounts of domain knowledge to control the search. They have so far been unable to place this type of knowledge in a general purpose planner.

Several specialized planners have arisen from the work in general AI planning. These alternative systems focus on a particular application of planning. They use techniques that work only for their given domain. Their advantage is that, for the given domain, these planners can be

very efficient. Our discussion will focus on those specialized planners that are most applicable to OMP.

3.0 ALTERNATIVE PLANNERS

The specialized form of planning that is most relevant to the OMP is scheduling. The emphasis in scheduling is on determining the plan steps' start and stop times and assigning needed resources. Schedulers, as a specialized form of planners, are able to perform well by capitalizing on problem-specific information. To accomplish this, an appropriate representation of the problem domain environment is essential. This representation must provide an easily utilized model of the constraint-driving aspects of the problem.

In less complex domains, the restrictions imposed by the problem constrain the set of admissible solutions to the extent that least commitment and constraint-propagation techniques are sufficient to reach an acceptable solution. In more complex domains, however, the scheduling restrictions leave the problem severely under-constrained. Since there are so many different alternatives at each point in the search, a strict policy of least-commitment and constraint-propagation will not succeed. This is especially true of interactions which occur when several different goals simultaneously request the same types of limited resources. It is also true in oversubscribed domains where a legal schedule cannot contain all the goals.

A more successful alternative to least commitment in highly under-constrained problems is to make commitments regarding specific scheduling decisions early in the search process. Using this philosophy, the control of the planning process becomes the problem of determining the order in which to make commitments or, alternatively, determining which constraints should be relaxed. Managing the complexity associated with this approach to real-world problems depends upon effective utilization of knowledge about domain-specific constraints.

ISIS [18] is a planning system which works on job shop scheduling. ISIS models job shop scheduling as the allocation over time of a finite set of resources to specific manufacturing operations. Scheduling is viewed as a synthesis task in which it is infeasible to explore all viable alternatives. Therefore, "the role of constraints must be extended beyond the concept of 'winnowing' an enumerated set in order to reduce the combinatorics of the search process."

The planning process in ISIS is divided into two parts: determining the appropriate sequence of steps and assigning the required resources and time intervals to the steps selected. The scheduling is a constraint-driven process which has two classes of constraints. The first class of constraints is scheduling restrictions that serve to delineate the space of possibilities in developing a schedule. The second class of constraints provides scheduling preferences for differentiating among possible schedule choices.

ISIS uses a hierarchical search paradigm. The first abstract level is based on resource capacity. This produces time-bound constraints for the lower levels. In the detail levels these time-bound constraints provide a "periscope effect" on the search. They allow the local detail search to consider more global effects of a commitment.

ISIS explores alternative schedules based on a beam search paradigm. At each iteration in the search process, ISIS retains and extends only the N best partial schedules. Constraints and preferences are used to differentiate among alternatives. Constraints leading to a decision are maintained to restrict the final determination of a particular order's schedule.

There are various perspectives that a planner can use in building a schedule. ISIS uses an order-based perspective. In an order-based perspective the planner takes an order and assigns the time slot and resources to that order before working on another order. On the other hand, a resource-based perspective views scheduling as a collection of resources. The system restricts the scheduling process into developing a single resource at a time. Since ISIS uses an order-based perspective, it cannot make use of tactics that try to resolve resource bottlenecks before trying to resolve conflicts in the other resources.

OPIS [34] is a job shop scheduler that grew out of the experience with ISIS. OPIS implements multiple scheduling perspectives. It incorporates the order-based perspective used in ISIS and adds a resource-based perspective. The resource-based perspective is used to focus the search on the resource bottlenecks. After the resource bottlenecks have been scheduled, OPIS shifts to an order-based perspective and finishes scheduling the remaining orders.

The current implementation of OPIS uses a blackboard-style architecture. The system organization consists of knowledge sources (KS) which implement alternative scheduling strategies and maintain the current schedule hypotheses. A resource-based or order-based perspective is provided by centralized blackboard management and event-based control. The centralized blackboard manager is solely responsible for maintaining an accurate description of the current state of a candidate schedule. The event-based control manager controls the problem decomposition and coordination of the scheduling process in an opportunistic fashion. A hierarchy of event types provides a basis for structuring the event's control knowledge. This knowledge is used to focus the search.

Within its control structure, OPIS is able to perform resource-based and order-based scheduling. It has a capacity analyzer which generates predictions of likely areas of high resource contention (bottlenecks). This function provides the basis for dynamic search control and enables OPIS to exploit the resource-based scheduling perspective.

Over the years many algorithms for scheduling have been developed. Some of the most popular are dynamic programming, hill climbing, and simulated annealing. All these algorithms contain the concept of evaluating a schedule and trying to optimize this evaluation.

Dynamic Programming was developed as a part of operation research. It is guaranteed to find an optimal solution. It is also guaranteed to be an NP-complete search. Dynamic programming can also be used to find nearly optimal solutions with less computational effort. The search is terminated when a solution is found in a specified range.

A simpler technique is the hill climber. A hill climber takes a schedule and modifies one of its parameters at a time. If this causes the evaluation of the schedule to improve, then the modified schedule becomes the new master schedule. If the evaluation function is thought of in

terms of hills and valleys on a space of all possible schedules then a hill climber simply starts at some point and climbs until it reaches some maximum point. This point is a locally optimal schedule but not necessary a globally optimal schedule.

The simulated annealer technique derives its name from the software simulation of the crystallization of metals. When an atom crystallizes, it tries to achieve a local energy minimum much like a hill climber. During the annealing process, however, the thermal energy adds a random value to the energy. This is simulated in software by adding a random value to the evaluation of the system. As the metal cools, this random value decreases until the system becomes a simple hill climber. The advantage of a simulated annealer is that it can find a more global maximum than a hill climber. It is not, however, guaranteed to find the optimal solution.

None of these techniques used alone can do a realistic job in scheduling large problems. It is, however, common to combine these techniques with control heuristics. These heuristics try to direct and limit the search that is performed by one of these techniques.

There are many control heuristics for scheduling derived from operation research. Many of these heuristics try to make predictions about the future state of the schedule so that current decisions will avoid future bottlenecks. In making an effective local commitment, it is necessary to have some prediction of its global impact on the plan both now and in the future development of the plan. Some schedulers such as RALPH (Resource Allocation Planning Helper) [24] and OPT (Optimize Production Technology) [26] accomplish this by performing two passes over the schedule. Each of the two passes starts with the goals and builds a new schedule. The first pass uses a simple scheduling algorithm and ignores some types of conflicts. The intent is to gather information about the resource capacity of the schedule. This resource capacity schedule is then used to make predictions about which events and time regions will cause the most problems. The second pass then starts from scratch and rebuilds the schedule again by using a more sophisticated technique. The planner uses the predictions generated by the first pass to make its scheduling commitments in such a manner as to avoid the problem areas found in the first pass.

Both RALPH and OPT use a resource capacity scheduler for their first pass. RALPH then uses a dynamic program for the second pass. The evaluation function tries to minimize conflicts with the steps that have been scheduled and tries to minimize overlap with high use areas identified by the first pass. In both the first and second pass RALPH uses an order-based perspective. After it has finished its first pass, OPT identifies the bottleneck resources. OPT's basic assumption is that there are generally five or six bottleneck resources, but which resources are the bottlenecks depends on the particular schedule. OPT then uses a simulated annealer to schedule the bottleneck resources. After the resource bottlenecks have been scheduled, OPT uses a resource capacity scheduler, much like its first pass, to finish up the remaining orders. Thus, OPT's second pass uses both a resource-based perspective and an order-based perspective. By using this two-pass approach both RALPH and OPT have been used to solve real-world scheduling problems.

Plan-It (Plan Integrated Timelines) [3] is a multi-pass scheduler. The tasks are first scheduled using a simple resource loading technique. Then Plan-It has a set of hill climbers that

can be invoked on the schedule. Each of these hill climbers focuses on a different aspect of the schedule. The selection of the hill climber is left to the human expert running the system.

4.0 SUMMARY

The most advanced planners discussed are FORBIN, OPIS, and OPT. These planners combine several different approaches to planning. No one approach can, by itself, succeed in any large domain. Both OPIS and OPT divide the resources into bottleneck and non-bottleneck resources. They then use different planning techniques on each of these classes of resources. However, both these systems must perform some search before they can successfully divide the resources.

All three planners use heuristics to try to contain the search. However, just using heuristics is not enough. The heuristics cannot see far enough into the future devolvement of the schedule. FORBIN and OPIS use hierarchical planning to develop a high-level abstract plan before doing the detail plan. FORBIN has a scheduling module which alerts the planner to a bad decision early in the search process. OPIS, also includes a pre- and post-search which identify possible problems. OPT carries this idea to include an independent first-pass search. To know which scheduling decision to make, one needs to have already created the schedule.

Planning research started out with general purpose planners and planners that found optimal schedules. These planners could not solve large real-world planning problems. The search space grows too fast, and not enough is known about general purpose search control. The current research focuses on adding domain-specific knowledge to control the search. These new planners combine several different techniques in a effort to add this domain-specific knowledge.

Existing high-performance planners are specialized to a single specific problem domain. As the various planning techniques are combined and extended, one should see the emergence of multi-domain planners that are capable of handling real-world problems.

REFERENCES

[1] Allen, J. F., *Maintaining Knowledge About Temporal Intervals*, Communications of the ACM, Vol. 26, No. 11, pp. 832–843, 1983.

[2] Bell, Colin, *Resource Management in Automated Planning*, AIAI TR-8 Artificial Intelligence Applications Institute, University of Edinburgh, 1985.

[3] Biefeld, Eric, PLAN-IT: *Knowledge-Based Mission Sequencing*, Proceedings of SPIE on Space Station Automation, pp. 126–130, October, 1986.

[4] Chapman, D., *Nonlinear Planning: A Rigorous Reconstruction,* Proceedings of IJCAI-9, pp. 1022–1024, 1985.

[5] Cheeseman, P., *A Representation of Time for Planning*, AI Center, SRI, Menlo Park, Ca. Tech. Note 278, Feb. (1983).

[6] Currie, K., and A. Tate, *O-Plan: Control in the Open Planning Architecture,* Proceedings of the BCS Expert Systems '85 Conference, 1985.

[7] Dean, Thomas, and Drew McDermott, *Temporal Data Base Management*, Artificial Intelligence Journal, Vol. 32, No. 1, pp. 1–55, 1987.

[8] Dean, Thomas, R. James Firby, and David Miller, *The FORBIN Paper*, Yale Technical Report 433, Yale University, 1987.

[9] Dean, Thomas, *Intractability and Time Dependent Planning*, Workshop on Planning and Reasoning About Action, 1986.

[10] Fikes, R. E., and Nils Nilsson, *STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving*, Artificial Intelligence Journal, Volume 2, pp. 189–208, 1971.

[11] Friedland, Peter, *Knowledge-Based Experiment Design in Molecular Genetics*, Report No. 79-771, Computer Science Dept., Stanford University, 1979.

[12] de Kleer, J., *Choices Without Backtracking*, Proceedings of AAAI-84, pp. 79–85, 1984.

[13] Drummond, Mark, *A Formal Representation of Action and Belief for Automated Planning Systems*, Workshop on Planning and Reasoning About Action, 1986.

[14] Drummond, Mark, *Refining and Extending the Procedural Net*, Proceedings of IJCAI-9, pp. 1010–1012, 1985.

[15] Ernst, George and Allen Newell, *GPS: A Case Study in Generality and Problem Solving*, Academic Press, New York,1969.

[16] Fikes, R. E., *Knowledge Representation in Automatic Planning Systems*, A. K. Jones (ed.), Perspectives on Computer Science, Academic Press, New York, 1977.

[17] Fox, Mark S., B. P. Allen, and G. A. Strohm, *Job Shop Scheduling: An Investigation in Constraint-Directed Reasoning*, Proceedings of the Second National Conference on Artificial Intelligence, pp. 155–158, 1982.

[18] Fox, M. S. , *Constraint-Directed Search: A Case Study of Job-Shop Scheduling*, Ph.D. Dissertation, Computer Science Dept., Carnegie-Mellon University, Oct., 1983.

[19] Glover, Fred, and Claude McMillan, *The Combinatoric Explosion: A Reassessment of Impacts of Microcomputers on Operations Research*, Elsevier, 1986.

[20] Hayes-Roth, Barbara, *A Blackboard Architecture for Control*, Artificial Intelligence Journal, Vol. 26, pp. 251–321, 1985.

[21] Hayes-Roth, Barbara, Frederick Hayes-Roth, Stan Rosenschein and Stephanie Cammarata, Modeling Planning as an Incremental, Opportunistic Process, Proceedings of IJCAI-6, pp. 375–383, 1979.

[22] Hayes-Roth, Barbara and Hayes-Roth Frederick, *A Cognitive Model of Planning*, Cognitive Science, Vol. 3, pp. 275–310, 1979.

[23] Hayes-Roth, Barbara and Frederick Hayes-Roth, *Cognitive Processes in Planning*, Rand Corporation Report R-2366-ONR, 1987.

[24] Johnson, Craig, and David Werntz, *Automation of the Resource Allocation and Planning System at NASA's Jet Propulsion Laboratory*, Proceedings of SPACE: Technology, Commerce and Communications, Houston, Texas, November, 1987.

[25] Le Pape, Claude and Stephen Smith, *Management of Temporal Constraints for Factory Scheduling*, Proceedings of the Working Conference on Temporal Aspects in Information Systems, 1987.

[26] Meleton, Marcus P., Jr., *OPT-Fantasy or Breakthrough?*, Production and Inventory Management Vol. 27, No. 2, 1986.

[27] Minifie, J. Roberta, and Robert Davis, *Survey of MRP Nervousness Issues*, Production and Inventory Management Vol. 27, No. 3, 1986.

[28] McDermott, D. A., *Planning and Acting*, Cognitive Science, Vol. 2, No. 2, pp. 71–109, 1978.

[29] McDermott, D. A., *A Temporal Logic for Reasoning About Processes and Plans*, Cognitive Science, Vol. 6, pp. 101–155, 1982.

[30] Porta, Harry, *SWITCH Users' Manual*, JPL Publication 86-24, Jet Propulsion Laboratory, California Institute of Technology, 1987.

[31] Sacerdoti, Earl, *Planning in a Hierarchy of Abstract Spaces*, AI Vol. 5, No. 2, pp. 115–135, 1974.

[32] Sacerdoti, Earl, *A Structure for Plans and Behavior*, New York, Elsevier North-Holland, Inc., 1977.

[33] Sacerdoti, Earl, *Problem Solving Tactics*, IJCAI, pp. 1077–1085, 1979.

[34] Smith, Stephen, Mark Fox, and Peng Si Ow, *Constructing and Maintaining Detailed Production Plans: Investigations into the Development of Knowledge-Based Factory Scheduling Systems*, AI Magazine, Vol. 7, No. 4, 1986.

[35] Smith, Stephen, Peng Si Ow, Claude Le Pape, Bruce McLaren and Nicola Muscettola, *A Constraint-Based Framework for Reactive Management of Factory Schedules*, Proceedings of the International Conference on Expert Systems and the Leading Edge in Production Planning and Control, pp. 349–366, 1987.

[36] Stefik, Mark, *Planning with Constraints (MOLGEN: Part 1)*, Artificial Intelligence Journal, Vol. 16, No. 2, pp. 111–140, 1981.

[38] Stefik, Mark, *Planning and Meta-planning (MOLGEN: Part 2)*, Artificial Intelligence Journal, Vol. 16, No. 2, pp. 141–169, 1981.

[39] Tate, Austin, *Project Planning Using a Hierarchical Non-linear Planner*, Department of Artificial Intelligence Report No. 25, Edinburgh University, 1976.

[40] Tate, Austin, *Planning with Multiple Resource Constraints and an Application to a Naval Planning Problem*, Proc. First Conf. on AI Applications, pp. 410–415, 1984.

[41] Tate, Austin, *A Review of Knowledge-Based Planning Techniques*, Knowledge Engineer's Review, Vol. 1, No. 2, British Computer Society Specialist Group on Expert Systems, June, 1985.

[42] Vere, Steven, *Planning in Time: Windows and Durations for Activities and Goals*, IEEE Transactions on Machine Intelligence PAMI-5, No. 3, pp. 246–267, May, 1983.

[43] Webb, W. Allan, *Combinatorial Complexity of the Flight Project/Deep Space Network, Resource Allocation/Scheduling Problem*, Operations Research Report, Jet Propulsion Laboratory, California Institute of Technology, 1986.

[44] Wesson, R. B., *Planning in the World of the Air Traffic Controller*, IJCAI, pp. 473–479, 1979.

[45] Wilensky, R., *Meta-Planning: Representing and Using Knowledge About Planning in Problem Solving and Natural Language Understanding*, Cognitive Science, Vol. 5, pp. 197–233, 1981.

[46] Wilkins, David E., *Domain-Independent Planning: Representation and Plan Generation*, Artificial Intelligence, Vol. 22, pp. 269–301, April, 1984.

APPENDIX G

REPLANNING AND ITERATIVE REFINEMENT IN MISSION SCHEDULING

Eric Biefeld
Lynne Cooper

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109

Presented at the Sixth Annual Intelligence Community AI Symposium, October 12-14, 1988

INTRODUCTION

Scheduling science experiments for such projects as Viking, Voyager, and Spacelab consumes a large amount of time and manpower. Whenever the Voyager spacecraft encounters a planet, the science experiments must be preplanned and ready to execute. This is a difficult scheduling problem due to the number and complexity of the experiments and the extremely limited resources of a spacecraft.

Since very few opportunities for space science exist, the major goal of mission scheduling is to maximize the number of science experiments that can be performed using the limited resources of the spacecraft. The total amount of requested experiments can be several times the amount that the project can accomplish.

Not only are schedules oversubscribed, they are also dynamic. Although the Voyager spacecraft was built and launched years ago, the flight rules governing the use of the spacecraft have changed. As the scientists learn more about their objectives, the experiment requests are updated. Thus, the mission schedule is a dynamic entity.

The Jet Propulsion Laboratory (JPL) has performed mission scheduling for many years with a variety of deep space flight projects. The effort in scheduling an entire project such as Voyager can be measured in man-centuries. Because of this huge cost, JPL has been researching advanced software-scheduling systems for several years (e.g., Deviser, Plan-It, Switch, Ralph).

Our current research, the Operations Mission Planner (OMP), is centered on minimally disruptive (non-nervous) replanning and the use of heuristics to limit the scheduler's search space. This paper addresses some of the problems pertinent to mission scheduling. It then defines iterative refinement, one of the basic design goals of our current research. This work has been greatly influenced by discussions with, and the observations of, the expert mission schedulers for the Viking, Voyager, and Spacelab projects.

MISSION PLANNING

Mission planning, for space systems such as Voyager, Viking, and Spacelab, is the process of scheduling large sets of science experiments to be conducted on board the spacecraft. This type of planning is a form of resource scheduling. The requested tasks consist of the science experiments, while the resources are the various components of the spacecraft needed to perform the science experiments. The Mission Sequence Team must devise a schedule which achieves the maximum science return.

An experiment is accomplished by executing a series of steps on board the spacecraft. Each of these steps utilizes several of the resources available on the spacecraft. Temporal relationships exist between the steps which compose a task and between these steps and the absolute time of the mission. This includes precedence ordering between the steps (as well as the absolute time) and windows of opportunity for the task. There may also exist some resource constraints between the steps of a task. For example, a task may require that the same astronaut perform all the steps in a single experiment aboard Spacelab.

Legal task decompositions are provided in advance to the scheduler. For Spacelab missions, the tasks are submitted to the scheduling team already decomposed into the legal series of steps. For Voyager, two or three different series of steps may exist that could accomplish a task, but these alternative task expansions are known in advance of the scheduling. The scheduler determines when a task will occur, which of the possible resources it will use, and the global utilization of the various spacecraft resources through time.

The sheer size and complexity of a spacecraft mission makes mission scheduling a difficult process. The Voyager spacecraft's near encounter with Uranus (the phase of the encounter at the closest approach to the planet) was comprised of approximately 175 different experiments conducted during a 48-hour period. The initial mission sequence was generated by a team of 30 people over a six-month period. An additional two to three months was spent modifying this initial mission sequence.

The resources of all science-gathering spacecraft are highly oversubscribed. The number of requested tasks greatly exceeds the spacecraft's abilities. There are simply too many important experiments to be accomplished using the limited resources of the spacecraft. As a result, many of the requested tasks are not included in the schedule. Once developed, the schedule is critiqued by the mission scientists based on the total science return which can be accomplished using that sequence. Because of the high cost of the spacecraft mission and the limited opportunity for planetary encounters (e.g., the Voyager-type Grand Tour of the solar system can be done only about once every 175 years, due to the unique alignment of the planets that makes it possible), the goal of the entire scheduling process is to maximize the total science returned.

Mission scheduling is a highly underconstrained planning problem. There exist many different, legal schedules that can satisfy the same set of requested tasks. The most important criteria used to judge the schedule are its impact on the health of the spacecraft and the total science returned. The mission scheduling team's main concern is how to achieve as much as possible in the limited time and limited resources of a mission. The schedulers do not perform an exhaustive check of the legitimacy of the schedule. Other mission specialists verify the schedule and use a large set of detailed simulations to test it. These specialists often request changes to the schedule to protect the health of the spacecraft and ensure the success of the mission.

Classical optimization techniques are not able to handle scheduling in this large complex domain. Advance scheduling such as replanning, heuristic planning, and iterative planning are used by expert human schedulers to support mission planning. Techniques for formalizing these concepts and incorporating them into an automated mission-planning system are needed and form the major thrust of this research.

REPLANNING

Since the world is not a static place, replanning is a functional requirement for scheduling. Events in the real world change the assumptions upon which a plan is based. These

events can be spectacular. For example, the first pictures returned by Voyager of Jupiter's moon, Io, showed a volcanic eruption. The mission scientists immediately requested changes in Voyager's schedule to obtain more information on this totally unexpected event. Most events are, however, more mundane and happen well in advance of the encounter.

During the flight, the spacecraft engineers constantly monitor the spacecraft. Depending on how the spacecraft degrades during the long flight, they will modify the flight rules governing the use of the spacecraft resources. These modifications are intended to protect the health of the spacecraft and ensure the success of the mission. The modification of the flight rules governing the use of the spacecraft's resources will, in most cases, require modifications to the schedule.

A currently popular approach to automated replanning is to simply plan again. The knowledge base and input tasks are updated and the software scheduler is rerun. The software scheduler then produces a new schedule which accomplishes the new tasks using the modified resources. Each time the scheduler runs, however, a radically new schedule is produced.

This approach leads to nervous replanning. This nervous behavior arises due to the underconstrained nature of the scheduling problem. For any mission scheduling-type problem, there exist many acceptable solutions that are radically different. Any change, however slight, in the planner's inputs may cause the planner to explore an entirely different section of the solution space. This change in the search will, most likely, lead to a schedule radically different from the original schedule. Mission planning has been shown to be extremely input-sensitive.

If the schedule were the final output of the system, nervous replanning might be tolerable. But the mission-planning schedules are an input to many other processes. In mission operations, a schedule is an input to a large evaluation and verification process. The verification and simulation of the schedule to guarantee that there are no hidden interactions that might jeopardize the health of the spacecraft may take several man-years.

For a scheduler to survive in an operational environment it must be capable of making small changes to an existing schedule. If the inference engine must do extensive backtracking in order to change a task, then the scheduler is destined to exhibit nervous replanning. The old schedule must therefore be an input to the scheduler. The scheduler knowledge base must include the operational cost of making a change to the existing schedule, and the scheduling inference engine must accommodate this operational requirement for non-nervous replanning.

HEURISTIC PLANNING

In spacecraft scheduling, the three major sources of planning knowledge are the mission scientists, the spacecraft engineers, and the mission schedulers. The scientists define the tasks, the relative priority of the tasks, and how the tasks can be achieved using the resources of the spacecraft (task decompositions). The engineers set up the flight rules defining the spacecraft resources and interactions between the resources. They create global rules and preferences for scheduling the spacecraft based on their knowledge of the underlying physical and operational constraints. For example, one rule may require ten minutes of quiet time for every two hours of continuous activity. The rules may be soft. For example, a scheduler may reduce the quiet time

to six minutes (instead of the required ten minutes) in a two-hour segment to meet the needs of a high-priority experiment, but the next two-hour segment must have its ten minutes of quiet time.

The other major source of knowledge, the mission scheduler, provides metaknowledge on how to schedule and the process of scheduling, as opposed to direct knowledge about the tasks or the resources of the plan. This knowledge (usually encapsulated in heuristics) allows the scheduler to focus the search. Without this, the large search space for mission scheduling would not allow a solution to be found in a reasonable time.

This metaknowledge describes which tasks and resources to consider first, where and when to search for alternatives, and to what depth to extend the search. Capturing this metaknowledge in a software-scheduling engine places many requirements on the software task and resource models and on the inference engine scheduling process. The thrust of our current research is to define the representations and planning processes necessary to capture this type of metaknowledge.

Many expert heuristics assume that the scheduler knows which resources are the bottlenecks. The bottleneck resources vary over the duration of the schedule. For example, the data bandwidth to Earth may be critical at the end of a schedule, while electric power may be the bottleneck at the beginning of a schedule.

Along with bottleneck resources, the expert scheduler reasons about "difficult" tasks. These tasks include those that "seem" like they should fit into the schedule but take a large percentage of the scheduling effort to plan. The difficult tasks generally fall into the intermediate priority range because high- and low-priority tasks are either forced into the schedule or left out during earlier processes. These tasks do not cause a large resource conflict in the schedule nor do they fit into any obvious place on the schedule. They do, however, require most of the scheduler's time.

RESOURCE TIMELINES

The expert scheduler does not know a priori which resources will be the bottlenecks at any given time in the schedule. This information depends on the exact configuration of the resources and the requested tasks. To predict where the bottlenecks will occur, the scheduler uses some simple planning heuristics to build a first pass of the schedule. This first pass will contain many resource conflicts. Areas of high resource-conflicts are candidate resource bottlenecks.

In many types of planners, a conflict in the resources corresponds to a logical inconsistency in the plan representation. For these types of planners, a resource conflict must be resolved as soon as it is identified. In the planners that are currently being developed, the planner must be able to represent conflicts in the schedule. A conflict is noted on the resource timelines, and the inference engine can later use this information to locate the resource bottlenecks.

The expert schedulers track resource utilization with a set of timelines. There are separate timelines for each resource on the spacecraft. These resource timelines show the resource utilization and the steps requesting the resource. The timelines flag any resource conflicts.

We are working on software which will explicitly maintain these resource timelines. Each resource timeline is a temporal database which tracks the requesting steps and the total resource utilization. The resource timelines can be queried to find the various oversubscribed temporal regions. Whenever the parameters of a step are changed, the appropriate resource timelines are automatically updated. The steps which compose a task contain the intratask constraints. Therefore, it is not possible to modify a step in such a manner to make a task inconsistent with itself. However, tasks can be represented in a schedule which is inconsistent due to the resource conflicts.

The ability to represent conflicts is very important for mission scheduling. The initial schedule that is published usually contains resource conflicts. These conflicts arise from high-priority tasks which the scheduler does not have the authority to delete from the initial schedule. Instead, the scheduler notes these schedules which then go to high-level conflict resolution meetings. The ability to explicitly represent resource conflicts is also necessary to allow direct capturing of the expert human scheduler's meta-scheduling knowledge.

Since the tasks interact only through the resource timelines, in some sense the tasks are independent. It is possible to modify a previously scheduled task without backtracking or updating any other scheduled task. Modifying a previously scheduled task may cause some resource conflicts, but at certain stages of the scheduling process, that is acceptable. The scheduler has the ability to just note the conflicts for later stages of processing.

ITERATIVE REFINEMENT

Iterative refinement is a technique used by expert spacecraft schedulers. The scheduler first lays out the highly constrained tasks over which he has little or no control. This forms a background against which the rest of the scheduling is done. The scheduler then places the tasks which impact large portions of the schedule. These may, for example, be a series of tasks that have to be performed at exactly one-hour intervals over a large portion of the schedule. Any changes to this type of task would cause changes to most of the schedule. If the scheduler gets stuck trying to place such a task, he may elect to move it, but only as a last resort. Next, the scheduler positions the high-priority tasks, minimizing the number of conflicts. Finally, to complete the initial loading process, the scheduler places the remaining tasks on the schedule. If, at this point, some of the lower priority tasks do not fit easily, the scheduler may simply ignore them.

After the loading process is done, the schedule is 80 percent complete (in the sense that most tasks are in their final position on the schedule), although some resource contentions may still exist. The scheduler has only spent about 20 percent of the total scheduling time at this point. The scheduler will spend the remaining time trying to fit a few more tasks into the schedule and trying to resolve resource contentions.

Up to this point in the scheduling process, the scheduler has been task oriented. Now the scheduler becomes much more resource oriented. The scheduler focuses on the tasks which are causing resource contentions on a particular resource and in a particular time region. After this area is fixed, the scheduler moves to another. Using this type of planning, the scheduler iterates the schedule repeatedly, each time refining it a little more. After each pass through the schedule, the scheduler is willing to do a deeper search on any single task because the total number of tasks needing to be searched will decrease.

By focusing on just one area at a time the scheduler may fix a portion of the schedule just to cause conflicts when the next portion of the schedule is processed. After several iterations, a small set of tasks will circulate through the problem areas of the schedule. In this stage of scheduling, the scheduler once again becomes task oriented. The scheduler focuses on this small set of hard-to-place tasks and performs the deepest search. The scheduler addresses any chain reactions resulting from moving a specific task. In Voyager scheduling, this reasoning recurs about three levels down. In Spacelab science scheduling, the depth of cutoff is about four levels down. It is important to realize, however, that at this point the scheduler has a small list of tasks to try. The scheduler also restricts the impacted tasks to those that seem flexible.

In the final stage of processing, the scheduler looks for underutilized areas of the schedule. The scheduler checks the list of unscheduled tasks looking for a task that could use these resources. This unscheduled task will, most likely, not fit directly into the schedule without causing some conflicts. Otherwise, the task would have been scheduled earlier in the process. The scheduler tries to adjust some of the tasks in the underutilized areas in order to make room for the unscheduled task. This may involve a series of shifts, but since both the task and the underutilized areas have been identified, it is a tightly focused search.

The schedule is then evaluated by the mission scientists for its total science return. The scientists negotiate with one another and with the scheduling team about which tasks to include in the final sequence. The results of the negotiations must be reflected in the schedule. Therefore, the evaluation process following the generation of the initial schedule often results in requests to change the schedule, and hence the requirement for the replanning capability discussed earlier.

Iterative planning consists of a series of techniques. Each technique is responsible for a different aspect of the overall planning process. The first of these techniques roughs out the plan and identifies areas of high resource conflict. The later techniques use the knowledge of the resource conflicts to refine the plan and solve many of the schedule problems. The final techniques try to solve the last of the conflicts and "optimize" the plan.

By specializing the planning techniques, each technique can be made more efficient. For example, the first techniques will use shallow searches over a broad spectrum of tasks. Later techniques will use deeper searches, but the search will only be applied to a limited number of tasks. They will use knowledge about the particular schedule (i.e., the current resource conflicts, those tasks which have changed most often in the scheduling process) to constrain the search space. The techniques will employ either a shallow and broad search or a deep and narrow

search. If a planner must perform a broad and deep search, it will not be able to compute the schedule in any reasonable time.

CONCLUSION

This iterative planning approach to scheduling arose from attempts to heuristically control the search space of mission scheduling. The source of the heuristics were the human schedulers of Voyager, Viking, and Spacelab, who provided information on the stages of the scheduling process. Earlier stages are concerned with "roughing out" the schedule, placing most of the tasks, and identifying the troubled areas. Later stages then use scheduling heuristics to refine the existing schedule.

Most of these heuristics assume that the scheduler knows which resources are the bottlenecks and which tasks are causing the most difficulty for the scheduler. The best way to identify these critical resources and tasks is from the schedule produced by the earlier stages. In order to know what to try next, one must already know what the schedule will be like.

Iterative planning assumes that the information gained by earlier techniques can be used by the later techniques to constrain the search space. Iterative planning also assumes that the schedule will not be changed dramatically by the later techniques. These assumptions seem to hold for the mission-scheduling domain, which is extremely underconstrained. There exist many possible schedules for a single set of requested tasks. Two different human schedulers will produce two very different but equally acceptable schedules, given the same set of requested tasks. If, however, one human scheduler must modify another person's schedule, the basic structure of the schedule will not be modified. Therefore, expert schedulers normally perform non-nervous replanning.

Our research at JPL is centered on heuristic mission scheduling. By using resource timelines to represent the schedule and to build a control structure that is capable of supporting iterative planning, we hope to produce a scheduler that can perform non-nervous replanning.

APPENDIX H

SCHEDULING WITH CHRONOLOGY-DIRECTED SEARCH

Presented at AIAA Computers in Aerospace VII October 3–5, 1989 and
Published in *Proceedings of AIAA Computers in Aerospace*
(AIAA 89-3137) VII, Vol. 2, 1989, pp. 1978–1087

E. Biefeld
L. Cooper

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109

Abstract

The intelligent use of scheduling heuristics can enable the production of effective schedules in spite of the inherent intractability of scheduling in complex, real-world domains. In order to effectively use these heuristics, information is needed on the current state of the evolving schedule. One method to obtain this information is to use chronologies—limited histories of the scheduling process. The chronology-directed search is an important component of the heuristic approach to automated scheduling.

1.0 Introduction

The planning and scheduling requirements for NASA deep space missions are extremely time and labor intensive. For major events, such as the Voyager II encounter with Uranus in 1986, the generation of acceptable schedules takes work-decades of effort[1]. Difficulties in scheduling arise from the sheer volume of possible interactions among requested science tasks. The number of requested tasks far exceeds the capabilities of the spacecraft, resulting in highly oversubscribed spacecraft resources. The scheduling process is made even more difficult because the problem space is underconstrained. The constraints on the domain are insufficient to narrow down the scheduling choices; therefore, virtually limitless numbers of acceptable schedules are possible.

The inherent intractability[2] of the Voyager-type scheduling problem renders the classical approaches to automated scheduling ineffective. Our research has centered on using powerful control heuristics to guide the search. Most of these control heuristics need knowledge of the schedule to be effective. For example, the identification of the most highly oversubscribed resources is critical in guiding the search process. The approach presented in this paper uses a history of the planning process to assist in identifying this type of knowledge about an evolving schedule.

The research presented here is based on the Operations Mission Planner (OMP) project. The OMP is an iterative planner which makes a series of passes over the schedule. Each pass further refines the schedule by performing a deeper, but more narrowly focused, search. The purpose of these iterations is to use the information gathered during the previous passes to guide the current pass. This information is kept in a variety of data objects, *chronograms*, referred to collectively as the *Chronology*.

In this paper, we present first an overview of heuristically controlled iterative scheduling. We then define chronologies and the role they play in the OMP approach. Next, we discuss how chronologies are being implemented and the impact this approach has on the OMP architecture. Finally, we present some future research issues.

2.0 Overview of Heuristically Controlled Iterative Refinement

Iterative refinement is a technique based on the methodology used by expert human[3] spacecraft schedulers. Human schedulers construct a schedule by first building a framework of the schedule. This framework is analyzed and refined by a series of different planning techniques[4]. The human schedulers work in phases—using different techniques in each phase, depending on

their assessment of the schedule. These phases are summarized in Figure 1 and are discussed in the following sections.

2.1 Initial Loading Phase

The scheduler first lays out the highly constrained tasks and the tasks which impact large portions of the schedule. The latter may, for example, be a series of tasks that have to be performed at exactly one-hour intervals over a large portion of the schedule. Any change to this type of tasking causes changes to most of the schedule. In the latter phases, the scheduler may elect to move these tasks, but only as a last resort. This forms a skeleton against which the remaining scheduling is done. Next, the scheduler positions the high-priority tasks, minimizing the number of conflicts. Finally, to complete the initial loading process, the scheduler places the remaining tasks on the schedule.

During the initial load, the scheduler is using simple techniques to place tasks in the schedule. The goal for this phase is to build up an initial schedule which is refined by later scheduling phases. The scheduler is not concerned with creating a conflict-free schedule at this point, although some of the simpler conflicts may be resolved during this phase.

Phase	Input	Focus	Heuristics	Output
Initial Load *Produce Initial Draft Schedule*	List of Tasks Resource Descriptions	Tasks *Shallow* *Entire Schedule*	Very Fast Very Simple Moves	Initial Schedule
Conflict-Centered *Identify Bottlenecks*	Initial Load Schedule	Resources *Shallow* *Individual Resource*	Simple Move Delete	Draft Schedule *80% in place*
Bottleneck-Centered *Classify Bottlenecks*	Draft Schedule	Bottlenecks *Deeper* *Time Intervals*	Complex Move Delete	Draft schedule *96% in place* *Conflict Free*
Optimization *Identify "should-fits"* *Opportunity search*	Draft Schedule List of DeletedTasks	Individual Tasks *Deepest* *Individual Task*	Complex Move Delete Interleave	Completed Schedule *Conflict Free*
Event Handler *Identify Planning Stage* *Initiate Replanning* *Minimize Disruption*	Executing Schedule Event	Event *Variable depending on Replanning Phase*	All of Above	Modified Schedule

Figure 1. Iterative Planning Phases

2.2 Conflict-Centered Phase

During the initial load process the scheduler has been *task* oriented. For the next two phases the scheduler becomes *resource* oriented[5]. In the conflict-centered phase, the scheduler will identify intervals on individual resources which have significantly high levels of conflict. The scheduler will resolve the conflict in these areas, making note of how this affects the other resources and time regions of the schedule. Using this type of planning, the scheduler reiterates on the schedule, each time refining it a little more and gaining additional information on interactions.

At the end of this stage, the schedule is 80% complete[6] (in the sense that most tasks are in their final positions on the schedule), although some resource contentions still exist. The scheduler has spent approximately 20% of the total scheduling time. The scheduler will spend the rest of the scheduling time trying to resolve the remaining resource contentions and fit a few more tasks into the schedule.

2.3 Bottleneck-Centered Phase

At the end of the conflict-centered phase, the scheduler has identified the bottleneck regions — time intervals on various resources with high levels of contention and substantial resource interaction among the involved tasks. These bottlenecks become the focus of future scheduling efforts. Rather than search over the entire schedule, the scheduler can concentrate its efforts on a group of small sections of the schedule.

By focusing on just one bottleneck at a time, the scheduler may apply a deeper search without spending exorbitant amounts of time. The scheduler will use the chronology to classify the bottlenecks according to size, complexity, and total amount of oversubscription. Depending on the classification of the bottleneck, the scheduler will use different heuristics to control the search. An example of this process is given in Section 3.0. At the end of this phase the schedule is conflict-free.

2.4 Optimization Phase

In the final phase of constructing a schedule, the scheduler once more becomes task oriented. After several iterations, a small set of tasks will have circulated through the problem areas of the schedule. There is a high probability that these tasks were deleted late in the bottleneck-centered phase. The scheduler focuses on this small set of *hard-to-place* tasks and performs the deepest search. The control heuristics will direct the search to regions of the schedule where little effort has been previously applied.

Unlike the earlier phases, the search immediately focuses on any conflict which results from modifying a task. Thus, if modifying a task causes a conflict, the search will continue by trying to modify the other tasks involved in the new conflict. The control heuristics restrict the depth so the search will progress through only a small number of tasks.

The optimization phase does not produce an optimized schedule in the classical mathematics and operations research sense. Rather, optimization, in our context, refers to fitting in additional tasks after a conflict-free schedule has already been produced.

2.5 Event-Handling Phase

Schedules often must be revised during execution to reflect changes in the operating environment, such as equipment failure and changes in tasking. For example, when Voyager first returned images of Jupiter's moon, Io, showing unexpected volcanic activity, scientists generated new tasks to observe this phenomenon. Using the iterative, multiphase approach, the scheduler can react to these events by simply returning to the appropriate scheduling phase. If a few new tasks are requested, the scheduler would support those tasks by returning to the optimization phase and using existing knowledge about the schedule to fit in the relatively small number of tasks.

If, however, the event is a catastrophic failure of an important, limited resource, then the scheduler would be required to make more drastic changes. Because information relating to resource usage would no longer be valid, the scheduler would have to return to one of the earlier phases. The scheduler, however, would not change those portions of the schedule which do not interact with the failed resource.

2.6 Summary

Iterative planning consists of a series of scheduling phases. Each phase is responsible for a different aspect of the overall planning process. The first of these techniques roughs out the plan and identifies areas of high resource conflicts. The later techniques use the knowledge of the resource conflicts to refine the plan and solve many of the scheduling problems. The final techniques try to solve the last of the conflicts and add a few more tasks. Once the schedule is executing, changes are accomplished by reverting to the appropriate planning phase and making use of the information available on the schedule up to that point. During each phase, the scheduler cycles through its scheduling activities until it determines that a change in phase is appropriate, as shown in Figure 2.

By specializing the planning techniques associated with each phase, the techniques can be made more efficient[7]. For example, the first techniques use shallow searches over a broad spectrum of tasks. Later techniques will use deeper searches which are applied to only a limited number of tasks. They will use knowledge about the particular schedule (i.e., the current resource conflicts, which tasks have changed most often in the scheduling process) to constrain the search space. The techniques will employ either a shallow and broad search or a deep and narrow search. If a planner must perform a broad and deep search, it will not be able to generate a schedule in any reasonable time. However, if the planner is always restricted to a shallow search, it will generate a severely suboptimal schedule.

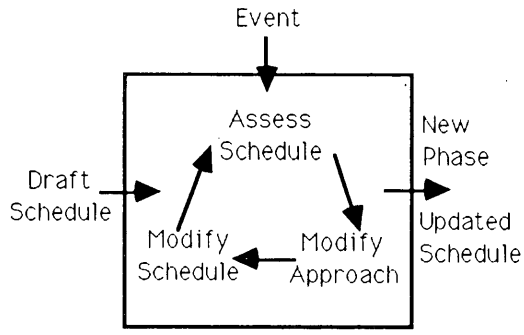


Figure 2. General Iterative Planning Cycle

3.0 What Is a Chronology?

A chronology is a limited history of the scheduling activity that has taken place. The chronology does not keep a complete snapshot of the changes taking place during the scheduling process. Rather, it focuses on characteristics which can provide information useful in directing subsequent searches. The chronology is used to identify interactions between time regions across several resources, detect the termination condition of a scheduling phase, and identify tasks that cause problems for the scheduler. Because we use an iterative approach to planning in which the scheduler focuses on either resources or tasks, the chronology keeps either resource or task information, depending upon the phase.

There are two activities associated with the chronology system: (1) collecting the information and (2) analyzing this information to characterize the schedule[8]. During the multiple passes of each scheduling phase, information is collected to help the scheduler identify when the goals for that phase have been accomplished. For example, during the resource-centered phase, the goal is to identify the bottlenecks. Information which enables the scheduler to determine the boundaries of the bottlenecks is collected and analyzed. Once the bottleneck areas have been identified, that phase is complete and the scheduler changes its focus to perform bottleneck-centered scheduling.

3.1 Bottleneck Identification Example

The identification of bottlenecks is an important and necessary step for effective scheduling. The exact location and extent of the bottlenecks are highly context-dependent[9]. Since the scheduler cannot anticipate where the bottlenecks will be located, the basic approach is to perform a simple exploration of the schedule space and use the information gathered to identify the bottlenecks.

After performing the initial expansion of the tasks into activities, the scheduler focuses on the area in the schedule with the most conflicts (Figure 3). The scheduler performs a shallow search, which lowers the number of conflicts in this area. Only the activities that are involved in the conflict are modified. The chronology module records the impact of these modifications on the resources.

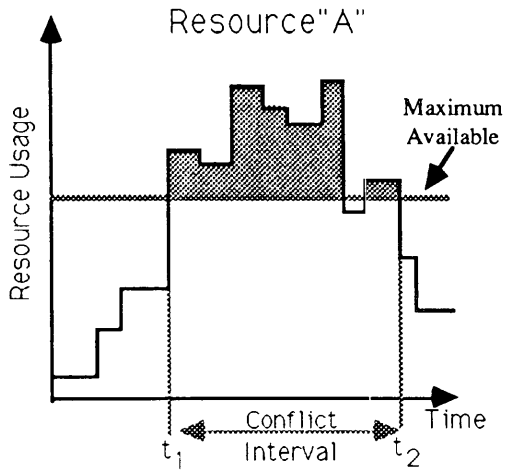


Figure 3. High-Conflict Area

While the search tries to avoid creating new conflicts, it will create them if necessary. The magnitude of these new conflicts may be larger than the magnitude of the original conflict that initiated the search. The scheduler will eventually focus on one of the new conflict areas. Solving this area may, in turn, cause other conflicts and so on, until the original conflict spot is once again in conflict. As the search progresses through the oversubscribed resources, the level of conflict in these and other areas oscillates. The conflict areas that continually oscillate in this manner are classified as potential bottlenecks.

As the scheduler focuses on a single conflict area, several other areas will be affected by the subsequent search. Since the conflict level for all these affected areas is modified during the same *focus state*, these areas and the conflict changes are all associated in the system's chronology. This chronological association of the oscillating resource areas allows the chronology module to group these areas into bottleneck regions.

3.2 Resource Bottlenecks

When one of the oscillating area's conflict levels is lowered, there will be several other areas whose conflict level is either lowered or raised by a similar amount. Those areas whose conflict level is simultaneously lowered are said to "oscillate in phase." These areas are usually located on the same resource and are related temporally. The resource areas which oscillate in phase are grouped together based on their chronograms to form *resource bottlenecks*, as shown in Figure 4.

3.3 Global Bottlenecks

After the chronology module has grouped the oscillating areas into resource bottlenecks, it tracks the coupling between the resource bottlenecks. When the conflict level of a resource bottleneck is lowered, several other resource bottlenecks will have their conflict level simultaneously raised. These resource bottlenecks, which oscillate "out of phase," represent different configurations of the same set of tasks. The ratio of the amount of conflict change between two resource

bottlenecks indicates the degree of coupling. *Global bottlenecks*, which consist of groups of resource bottlenecks that have a high degree of coupling, are identified based on the chronology. The scheduler will now enter the bottleneck-centered phase. Its efforts will focus on resolving the global bottlenecks, Figure 4.

3.4 Bottleneck Classification

Once identified, the chronology module classifies the bottleneck regions so the heuristics can adjust the scheduling strategy accordingly. If, for example, the average level of conflict is not much higher than the available resources in the bottleneck region, and it is tightly coupled and limited in extent, then the scheduler will perform a deep search on the activities in this bottleneck in order to optimize the schedule. If the bottleneck has a large average conflict level, then the scheduler will shrink the resource usage of the involved activities by deleting lower priority activities from the bottleneck.

Resource A has 3 high-conflict areas: **A1, A2, and A3**

Resource B has 3 high-conflict areas: **B1, B2, and B3**

Iteration	Active Conflict Area	A1	A2	A3	B1	B2	B3
1	A1	-2	-1	0	0	+1	+1
2	B2	+3	+2	0	-1	-4	-3
3	B3	+1	+1	0	0	-1	-2
4	A1	-4	-3	0	0	+3	+2
5	B2	+1	0	0	0	-1	0
6	A1	-3	-3	0	0	+3	+2
		Oscillating IN Phase				Oscillating IN Phase	

Oscillating OUT of Phase

A1 and A2 form a Resource Bottleneck: R-A1
B2 and B3 form a Resource Bottleneck: R-B1
R-A1 and R-B1 form a Global Bottleneck: G-1

Figure 4. Conflict Areas Oscillating IN and OUT of Phase

If, as another example, the bottleneck is loosely coupled and large in extent, then the search space is too large for the search module to directly find a good solution. The system will instead try to split the larger bottleneck into several smaller bottlenecks by deleting lower priority tasks. If this does not work, then the system will review the chronograms of the tasks in the bottleneck to identify those which affect a large portion of the bottleneck. These tasks will then be assigned to different regions of the bottleneck in such a manner as to allow the system to partition the one large bottleneck into a series of smaller, more manageable bottlenecks.

Scheduling is intractable; the scheduler must either perform an extremely large search or be willing, at times, to make arbitrary decisions. Locating and classifying the various bottlenecks can help to decompose the search[10] into a series of smaller, more manageable searches that can produce schedules which incorporate more tasks. At times, however, the scheduler must be willing to arbitrarily decompose the search.

4.0 Implementing Chronologies

The underlying assumption behind the use of chronologies is that scheduling actions that happen during the same planning phase are related and can give meaningful information about the state of the schedule. Our goal is to make use of this information to limit the search needed to find a highly successful schedule.

A chronology consists of a series of time stamps which associate a given state of the schedule with the actions that created that state. These time stamps, referred to as chronograms, are the basic building block in the development of a chronology. A chronogram uses the focus state of the scheduler as its identification. This state is a hierarchy of all the focus and subfocus states that the scheduler has implemented up to a given point.

For example, the scheduler will first set a general focus state, such as *roughing-out-schedule*. Under this state the scheduler will focus first on a resource, for example, *antenna-1*. Under this state the scheduler will set a subfocus on a particular interval of time, for example, *00:02:01 – 00:12:00*. Finally, the subfocus will be set to a particular task, *monitor-task-34*. The ID of the focus state is a list of the IDs of the different subfocuses that compose the current state. Thus, different entries into the chronology have chronograms that differ depending on the focus state.

4.1 Data Structure

The chronology is kept as a tree, as shown in Figure 5. The root of the tree represents the state *working-on-schedule*, while the next level down represents the major phases the system progresses through. Each successive node of the tree represents a new subfocus being added to the focus state, and the leaves contain the actions taken by the scheduler. When the chronology is analyzed, the information is taken from the appropriate subtree of the chronology that contains all the actions that occurred during the focus state of interest.

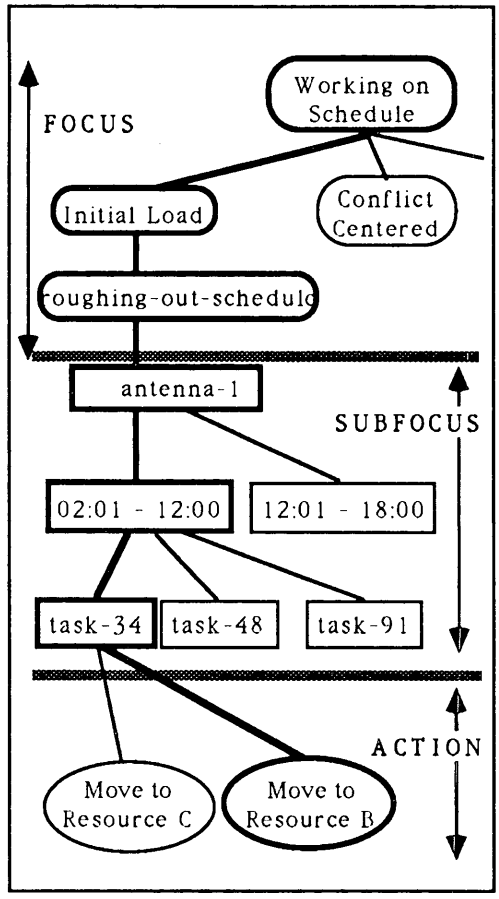


Figure 5. Chronology Tree
(Individual Chronogram in Bold)

4.2 Analysis

Analysis of the chronology leads to an abstraction of a subtree of the chronology. Figure 6 shows one possible abstraction of the tree presented in Figure 5. This abstraction states what was accomplished during the corresponding focus state. For example, the summation could state that most of the dense conflicts were easily solved, or it could state that many conflicts still exist after a simple search was used. This abstraction is spliced into the chronology, replacing the old subtree which specified the actions performed on the schedule. The use of abstraction relieves the severe burden that would result if all the information on the scheduling activities had to be retained. The pertinent information is available, in a usable and manageable format.

The use of the chronology also leads to the construction of new data structures, such as bottlenecks. These structures are linked to existing data structures, such as the resource timelines, as in Figure 7. The scheduler is then able to use these structures for future scheduling activities. The existence of these structures is also useful in determining the "goodness" of a particular schedule. It is easier to assess the state of the schedule at the higher level of abstraction provided by the new data structures and the chronology than by looking at the lowest level structures (timelines) in "raw" format.

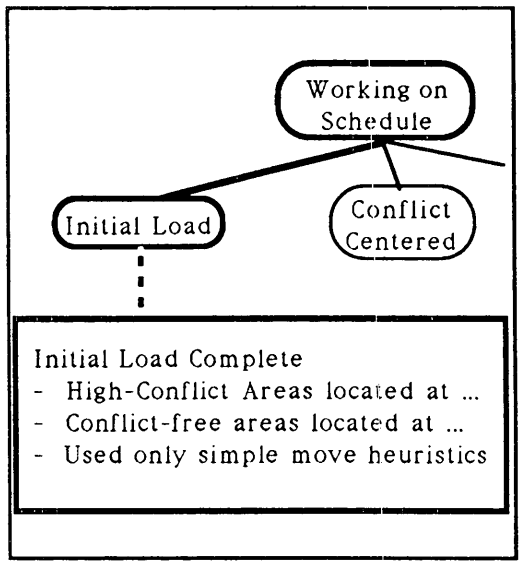


Figure 6. Abstraction of Chronology

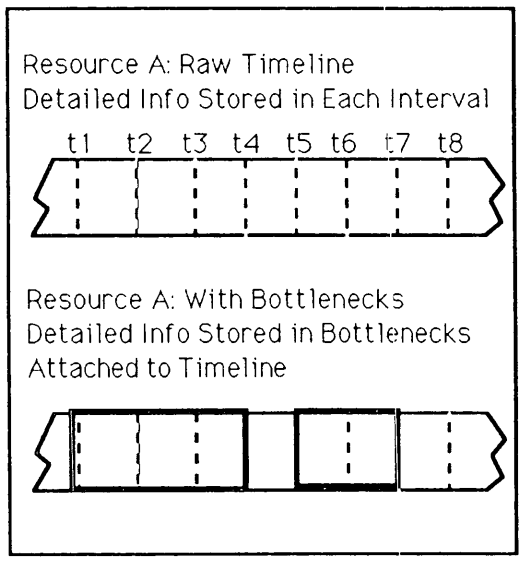


Figure 7. New Data Structures

5.0 Future Directions

A major concern in determining the usefulness of a chronology is the amount of information that must be kept and the amount of processing that must be done to the information to make it meaningful. The focus of our current research is on simple parameters and relatively simple algorithms for processing them. They have proved useful during the early planning phases, which are characterized by the use of simple heuristics and shallow search techniques. As we

progress to more advanced planning phases, which involve much deeper searches, the heuristics will become more complex, and we anticipate that the chronology information and algorithms will become more sophisticated and require the incorporation of domain-specific knowledge. The potential for improved performance, however, is greatest during these phases.

Scheduling research will continue to identify heuristics that can guide the scheduling process and the associated chronological information needed to support those heuristics. Currently, the early phases of the planning processes have been prototyped. Chronograms that identify the planning activity on a given task and the interactions between time regions on a given set of resources are included in the prototype. The prototype uses the chronology to identify the resource and global bottlenecks. The prototype then focuses its scheduling efforts on those bottlenecks. Future chronology work will focus on narrowing down the problem areas, providing more in-depth information on interactions, and characterizing the difficulty in scheduling a given task in more detail.

6.0 Conclusion

The major problem associated with automated schedulers for complex real-world domains is controlling the search. In domains such as Voyager mission scheduling, very detailed schedules are required. Unfortunately, without mechanisms to intelligently control the search, it is impossible to produce acceptable Voyager schedules: Either detail must be sacrificed or an exorbitant amount of time (centuries) must be spent developing the schedule.

There exist many different scheduling heuristics that focus the search on a particular aspect of the schedule. While these techniques exhibit excellent performance in some cases, they are not universally applicable. Therefore, the scheduler must identify when a particular scheduling heuristic may be appropriate. The iterative refinement approach is based on making the most effective use of the various scheduling heuristics.

In using the search, there is a trade-off between power and time; the deeper the search, the longer the time required. The use of a deep search over the entire schedule is infeasible and unnecessary, but limiting the deep search to limited segments where a less powerful search is ineffective is productive without incurring unreasonable costs.

The chronology system provides the necessary information for the control heuristics to determine which scheduling heuristics to use and where. This provides the scheduler with the flexibility necessary to approach the variety of scheduling problems encountered in the generation of a single schedule. This, in turn, enables the scheduler to expend a greater amount of effort on tightly focused areas, thus producing a more effective schedule. In the case of a Voyager-type project, where planetary encounters are once-in-a-lifetime events, the ability to fit just one more experiment into the schedule is a priceless opportunity.

Acknowledgements

In addition to the authors, individuals participating in this project include Loretta Falcone, Gaius Martin, and Kirk Kandt.

References

1 Biefeld, Eric (1986) "PLAN-IT: Knowledge-Based Missions Sequencing" *Proceedings of SPIE on Space Station Automation* pp. 126-130

2 Web, Allen (1986) "Combinatorial Complexity of the Flight Project/Deep Space Network Resource Allocation-Scheduling Problem" *Operations Research Report* Jet Propulsion Laboratory Internal Document

3 Hayes-Roth, Barbara, Federick Hayes-Roth, Stan Rosenschein and Stephanie Cammarata (1979) "Modeling Planning as an Incremental, Opportunistic Process" *Proceedings of the Sixth International Joint Conference on Artificial Intelligence* pp. 375-383

4 Biefeld, Eric and Lynne Cooper (1988) "Replanning & Iterative Refinement in Mission Scheduling" *Sixth Annual Intelligence Community AI Symposium*

5 Smith, Stephen and Peng Si Ow (1985) "The Use of Multiple Problem Decompositions in Time Constrained Planning Tasks" *Proceedings of the Ninth International Joint Conference On Artificial Intelligence* pp. 1013-1015

6 Biefeld, Eric, Lynne Cooper, Loretta Falcone, and Gaius Martin (1988) "Operations Mission Planner Annual Report," Jet Propulsion Laboratory Internal Document JPL D5685

7 Biefeld, Eric and Lynne Cooper (1989) "Comparison of Mission and Job Shop Scheduling" *Proceedings of the Third International Conference on Expert Systems and the Leading Edge in Production Planning and Control* pp. 483-494

8 Orelup, Mark, Paul Cohen, John Dixon, and Melvin Simmons (1988) "Dominic II: Meta-Level Control in Iterative Redesign" *Proceedings of the National Conference on Artificial Intelligence* pp. 22-30

9 Muscettola, N. and Stephen Smith (1987) "A Probabilistic Framework for Resource-Constrained Multi-Agent Planning" *Proceedings of the Tenth International Joint Conference On Artificial Intelligence* pp. 1063-1066

10 Johnson, Craig (1986) "A Look-Ahead Strategy for Heuristic Activity Scheduling" *Joint Conference of the Operations Research Society of America and the Institute of Management Sciences*

APPENDIX I

COMPARISON OF MISSION AND JOB SHOP SCHEDULING

Presented at

The Third International Conference on Expert Systems and the Leading Edge in Production and Operations Management

May 21–24, 1989

and
published in

The Proceedings of the Third International Conference on Expert Systems and the Leading Edge in Production and Operations Management,
Charleston, South Carolina, pp. 483–494

Eric Biefeld
Lynne Cooper

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109

COMPARISON OF MISSION AND JOB SHOP SCHEDULING

Eric W. Biefeld
Lynne P. Cooper
Jet Propulsion Laboratory
California Institute of Technology

ABSTRACT

This report compares the problem domains and solution techniques associated with mission and job shop scheduling. Although the domains are significantly different, they encounter many similar problems. Solutions identified for one domain have significant impact on the others. In order to institute a productive cross-fertilization of ideas and techniques from one domain to the other, it is important to recognize both the differences and similarities between mission and job shop scheduling.

1.0 INTRODUCTION

Mission scheduling, for space projects such as Voyager and Viking, shares many of the same fundamental problems as those encountered in manufacturing scheduling. While the problem domains are dissimilar, the goal for both domains is to accomplish the most tasks, using the least amount of resources, within the specified time constraints. Whether the task is manufacturing a widget or taking a photograph of the moons of Jupiter, the problem devolves into determining a sequence of steps which would accomplish the desired tasks.

Both mission and job shop scheduling face the primary problem confronting automated real-world scheduling systems: controlling the search process. While many of the operations' research-based optimization techniques perform well in limited domains, they quickly become bogged down in a combinatorial explosion when confronted with real-world complexity. Successful automation of scheduling is therefore dependent upon the development of powerful and flexible control methods.

Of particular interest is the application of Artificial Intelligence (AI) techniques. In general, both mission and job shop scheduling can be described as resource allocation problems in constrained environments. Techniques such as expert and knowledge based systems, and heuristic control show great promise toward controlling the search. Application of these AI techniques will prove beneficial for both mission and job shop scheduling.

This report first provides an overview of the mission and job shop domains. It then provides a summary of the similarities and differences of the domains and how these affect automation of the scheduling process. Finally, the report presents a discussion of the Artificial Intelligence techniques currently being evaluated for mission scheduling and how they could be used to support job shop scheduling.

2.0 OVERVIEW OF MISSION AND JOB SHOP DOMAINS

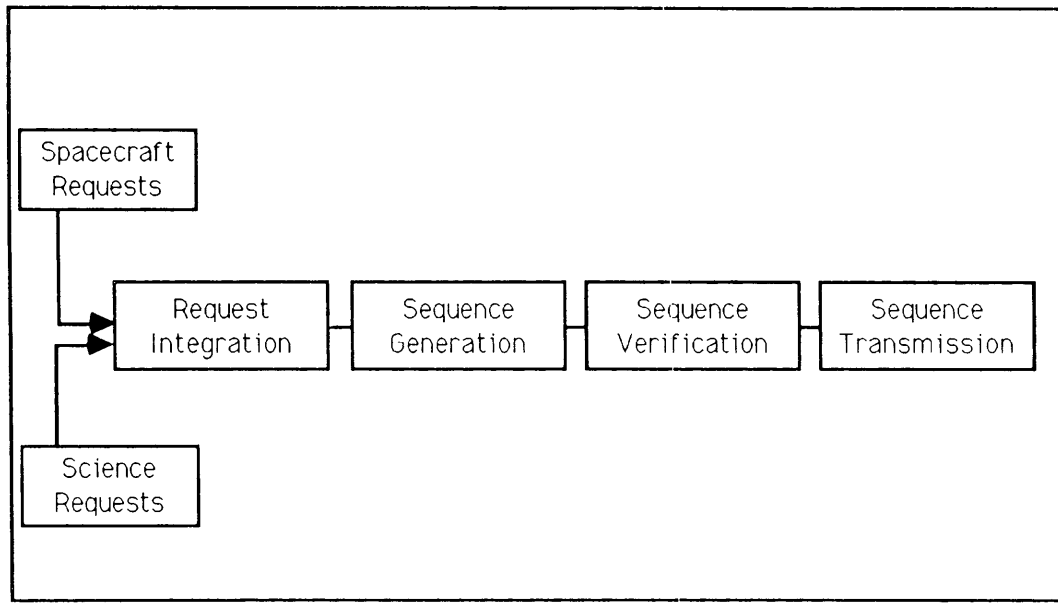
2.1 Mission Scheduling

Mission scheduling is the process of scheduling a large set of science experiments to be conducted on board a spacecraft. Tasking consists of requests for science experiments (take a picture of a specified phenomenon) and mission support activities (maneuver the spacecraft). The tasks compete for the use of the spacecraft subsystems [1]. The mission schedule consists of a sequence of commands which will be transmitted to the spacecraft. These commands operate the different spacecraft subsystems which are required to support the science experiments. Substantial effort is placed in verifying that the sequence will not harm the spacecraft.

In mission scheduling, the total number of tasks far exceeds the capabilities of the spacecraft. Therefore, many of the original science requests cannot be accomplished, and the primary goal of the mission schedulers is to achieve as many of the science requests as possible.

Expert human schedulers spend work-decades building and revising mission schedules for such events as the Voyager encounter with Uranus. Additional work-decades then go into the intense verification process. The schedule, however, is always subject to change during implementation due to the almost inevitable discoveries which are made during the mission. For example, while photographing Io, a moon of Jupiter, scientists discovered unexpected volcanic activity. The schedule had to be modified to handle the avalanche of high-priority requests to observe this phenomenon.

Figure 1. Mission Scheduling



The mission scheduling process, as shown in Figure 1, consists of: (1) receiving the high-level requests for science experiments as input, (2) breaking down these tasks into their component steps, (3) building a schedule and resolving any conflicts, (4) verifying that the schedule will not harm the spacecraft, and (5) changing the schedule as needed to support unusual occurrences or correct for spacecraft anomalies [2]. Throughout the entire process, the scheduler must concentrate on maximizing the science returned from the spacecraft.

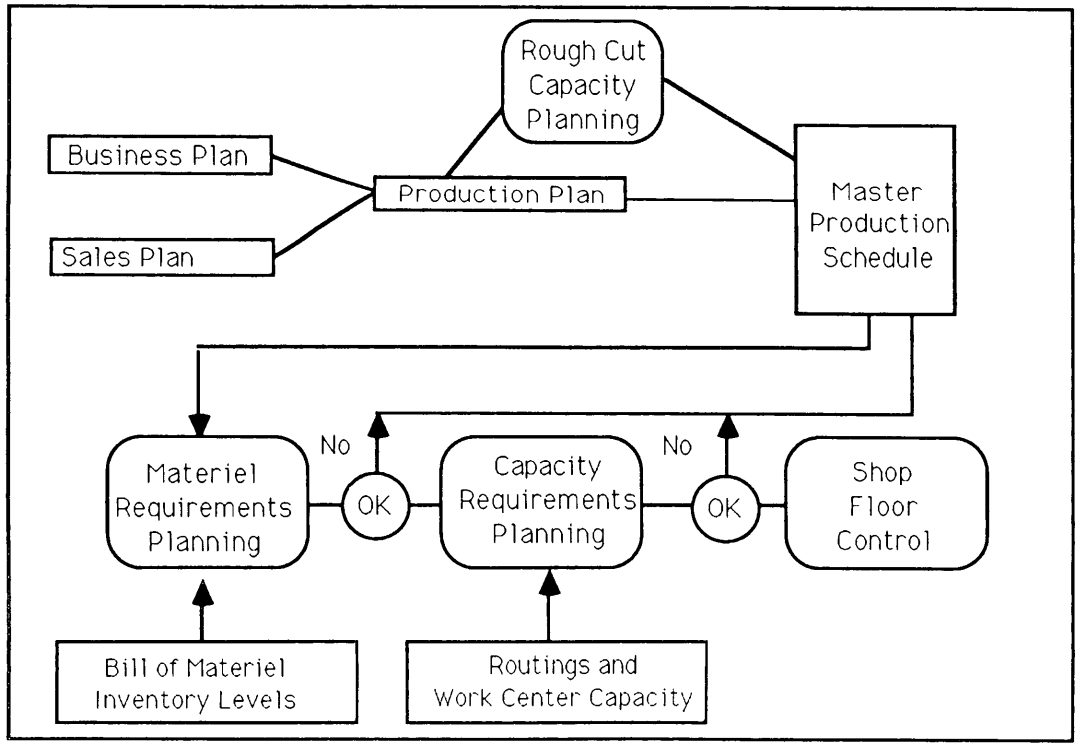
2.2 Job Shop Scheduling

Job shop scheduling is the process of scheduling the manufacture of commercial products. The inputs for job shop scheduling are the customer orders which compete for the use of equipment and materials. When complete, the schedule consists of a group of work orders and a group of purchase orders. These work orders are given to the shop managers on the factory floor who are then responsible for the production of the customer orders.

In job shop scheduling, there are multiple goals which must be satisfied. These goals include minimizing tardiness, maximizing resource utilization, and minimizing the work in progress. These goals are often competing, and the scheduling system must balance them when producing a schedule.

The job shop environment is a production environment. Schedules are automatically generated on a weekly or monthly basis and are maintained by hand in between. Once the schedule is produced, however, it is subject to change. Equipment breakdowns and rush orders will result in changes.

Figure 2. Factory Scheduling (MRP II)



In job shop scheduling, as shown in Figure 2, the schedule is usually developed through the interaction of several distinct planning modules [3]. The first module develops a "master schedule" from the business plan and the customer orders. This becomes the input to the MRP II (Manufacturing Resource Planning) system. The MRP II system then "explodes" the orders using the bill of materiel, checks inventory for the availability of the needed components, and then assigns release dates to the individual items in the order so as to achieve the final customer order on time. After the materiel planning, the system assigns resources to the proposed schedule. Since there may not be enough capacity to satisfy the schedule, the system will have to regenerate the materiel planning until an acceptable schedule can be found. The final MRP II schedule will be used to release orders to the shop floor where a dispatcher will assign the actual state times and resources to an order.

3.0 COMPARISON

3.1 Similarities

Mission and job shop scheduling domains share many characteristics. Their domains are similar in size and complexity. The tasks in both domains interact through resources. These resources also have similar characteristics which require constraint handling techniques. In both domains, the schedule is subject to changes while executing. But any change in the schedule could impact the other processes which depend upon the schedule. Attempts to automate the planning process have been made in both domains.

3.1.1 Size and Complexity

The mission and the job shop domains are similarly large in size and complexity. Simple techniques which work in less realistic domains are unable to support scheduling in these domains with reasonable response times. Both mission and job shop scheduling are underconstrained. The variety of options available to produce a legitimate schedule is large, and the limiting constraints only serve to partially reduce the search space. Although the underconstrained characteristic of each domain results from a different cause, the effect is the same: an extremely large and difficult to control search space [4]; [5].

3.1.2 Categories of Resources

The resources for both mission and job shop scheduling fall into three categories: capacity, consumable, and "state." Capacity resources are those which support up to a certain number of tasks at any one time. As tasks finish using the resource, it becomes available to support other tasks [6]. On a spacecraft, resources such as the camera, data transmitting bandwidth, and the electrical power fall into this category. A camera can only take one picture at a time, there is a limit on the amount of data that can be transmitted from the spacecraft, and, since the spacecraft generates all of its own electricity, the power consumption by the various subsystems on the spacecraft must remain within operational limits. Analogous resources in the job shop domain are the manufacturing equipment, such as the lathes and mills, and the skilled equipment operators.

Consumable resources are those which are not available after use. The consumable resources in the job shop scheduling domain are basically the raw materials used in the manufacturing process. These materials, however, are often handled by an MRP system and have associated lead times which must be considered during the scheduling process. Examples of these types of resources for the mission planning domain are the space available on a flight tape recorder or the propellant in the fuel tanks. Science data is often recorded on tape during a mission for transmission at a later time. The space available on the tapes is limited and must be expanded prior to recording additional information. Unlike manufacturing, mission planning sometimes has resources which cannot be replenished. For example, the propellant in the fuel tanks is a consumable resource which cannot be replenished.

The third category of resources is the state of the spacecraft or factory floor. Scheduling in both domains is sensitive to the state of the environment. For mission scheduling, the concerns include the orientation of the spacecraft, the direction of the various instruments, the level of vibration on the platform, and the release of any propellant which could interfere with an experiment. The state of the spacecraft environment may impose special constraints on the operation of other equipment required for a task.

On the factory floor, this is similar to determining what tool is currently in the lathe or the configuration of other pieces of equipment. The state of the equipment determines what additional steps, if any, are required to prepare the equipment for the execution of a task.

3.1.3 Interaction Through Resources

In both mission and job shop scheduling, tasks interact because of contention for the same resources. These interactions are referred to as conflicts. The interaction can be modeled in two basic fashions. In the classical planning systems (e.g., Noah, NONLIN, Deviser) [7]; [8], the schedule is modeled as an explicit task network. If an early task is changed, then the remainder of the schedule becomes invalid. Both the mission and job shop planning domains, however, can model the interaction between tasks as contention for resources. Interactions on capacity and consumable resources are limited to the availability or nonavailability of a given resource at a given time.

3.1.4 Reactiveness

Changes in the environment which affect the schedule occur in both the mission and job shop scheduling domains. Whether the change is the result of equipment breakdown/failure or the unexpected occurrence of a high-priority task, such as the opportunity to observe volcanic activity on Jupiter's moon, Io, or a rush order for a highly valued client, the scheduler must be able to accommodate these changes.

3.1.5 Schedules as Input to Other Processes

Scheduling in both domains is not a separate, independent task. Rather, the production of a schedule is just one step in the accomplishment of larger goals. The output of the scheduling process, the schedule itself, serves as input to other processes. For example, the mission

schedule must go through a long verification process prior to being executed. Therefore, any change in the schedule increases the effort expended on verification. Similarly, in manufacturing, the schedule is used as input for ordering materials, controlling inventory, and scheduling maintenance activities. Because impacts to the schedule also impact these types of activities, the way in which the scheduler reacts to changes could have far-reaching effects on the overall efficiency of the factory or spacecraft.

3.1.6 Replanning From Scratch vs Evolving (nonnervous)

In both domains, there are two major approaches to handling changes in the schedule. Replanning from scratch treats changes as if there were a new set of inputs to the scheduler and begins scheduling using this new set of inputs, from scratch. Scheduling, however, is highly input-sensitive, so replanning from scratch will generally result in an entirely different schedule [9]. An evolving schedule tries to maintain as much of the existing schedule as possible and modify (or evolve) [10]; [11] the schedule to accommodate the changes. Human schedulers naturally use the evolving schedule approach. Production types of automated schedulers in both domains have been limited to replanning from scratch. This is why MRP II systems are commonly run only on a monthly basis, and human schedulers are used to make changes between runs.

3.2 Differences

While there are substantial similarities between the mission and job shop scheduling domains, there are also subtle differences which must be taken into consideration when trying to generalize scheduling techniques to encompass both domains. The number of tasks, level of detail, acceptable cost, and the types of time and resource constraints are markedly different.

3.2.1 Oversubscription

In both mission and job shop scheduling, one of the most important goals is to accomplish as many of the requested tasks as possible while meeting any constraints on resources and time. When the number of tasks requested of the system is beyond those which it can physically support within the allotted time, the system is referred to as *oversubscribed.* On the whole, a factory is usually not oversubscribed throughout its lifetime, although there may be isolated instances when it is. A flight project such as Voyager, however, is massively oversubscribed. The science requests exceed the spacecraft capability by a factor of three.

When a system is oversubscribed, the only way to produce a schedule is to delete requested tasks. Until the scheduler can determine which tasks to delete, the scheduler must have some mechanism for representing schedules which have conflicts and must be capable of controlling the search in a solution space which contains these illegal schedules. This problem is a fundamental and inherent part of mission scheduling. Any common approaches among domains must be able to address this important aspect.

3.2.2 Level of Detail

Mission scheduling must be conducted on a substantially more detailed level than job shop scheduling. For example, a mission schedule will decompose tasks to the lowest level that can have any impact on the schedule. The start and stop times will be specified to the second. This type of detail is necessary because the schedule will be "compiled" into a command sequence which will be transmitted to the spacecraft. These commands cause the spacecraft to execute the various steps of the science tasking. If there are any unexpected interactions between the steps, then the health of the spacecraft will be in jeopardy.

Job shop scheduling is done at a much less detailed level. Most automated schedulers use an MRP II system which produces a schedule by using a queuing model of the job shop's resources. Exact start and stop times are not assigned to the tasks. Detailed automated schedulers, such as OPT [12], ISIS [13], and OPIS [14], will assign exact start and stop times but never to one-second accuracy. In a job shop, the schedule is used to dispatch work orders to the factory floor, which are then executed by humans. The schedule does not have to exactly specify each task, and there is enough flexibility to adjust to any unexpected interactions which may occur.

The spacecraft environment is substantially limited as compared with the factory floor. Although spacecraft missions do experience unpredictable changes, the mission environment is much more predictable than the factory floor. This makes the detailed scheduling of a spacecraft mission possible. While the factory floor is more likely to encounter unexpected events due to the environment, most factory floors are predictable enough that a more detailed scheduling approach is possible.

The level of detail used in mission scheduling makes several types of optimization of the schedule possible. For example, one of the most common optimizations is to interleave the steps from two or more separate tasks. This helps minimize the setup times of equipment that is shared by the activities. Another type of optimization is that the scheduler can opportunistically determine the lot size rather than use predetermined lot sizes. On the spacecraft, this corresponds to repeating an experiment a fewer number of times than requested by the scientist.

3.2.3 Amount of Time Spent Producing a Schedule

Another major difference in mission and job shop scheduling is the amount of time which can be spent to produce a schedule. Missions, such as Voyager, are extremely rare events, so the work-decades of effort used to produce a schedule are justified, and the time required to produce the schedule is available. Job shop scheduling, however, takes place on a routine basis. It would be extremely cost ineffective to focus the same intensity on production of the daily/weekly schedules as in mission scheduling.

Mission scheduling consumes many work-years of effort in a typical flight project. The 48-hour schedule of Voyager's near encounter with Uranus took 1.5 work-decades to produce. Because of the rare opportunity that such an encounter provides, substantial effort is expended optimizing the schedule to produce the maximum science return possible.

3.2.4 Temporal Constraints

Mission and job shop scheduling domains must address different types of temporal constraints. In the job shop domain, most tasks are assigned a specific due date. If it is impossible for the schedule to meet the due date, then the task is "slipped" and done at a later time [15]. In flight projects, a task is assigned to a time window [16], which is the temporal region during which the specific task can be performed. It is meaningless to perform the task outside of its associated time window.

In a mission such as Viking, a task will usually have a series of time windows where each corresponds to a different orbit of the spacecraft around Mars. Once the spacecraft is launched, it is physically impossible to substantially change these windows. Therefore, deadlines in mission scheduling are absolute and cannot be "slipped." If a task cannot be accomplished during its time window, the task will be dropped from the schedule. The ability to slip or drop tasks from the schedule substantially increases the complexity of the scheduling problem, underconstraining the scheduling problem, as addressed in Section 3.2.1.

3.2.5 Resource Constraints

In addition to specialized temporal constraints, the mission scheduling domain also must address special problems caused by the tight coupling of the scientific instruments on the spacecraft and the spacecraft support systems. Because the spacecraft is physically a closed environment, the effect of a given action must be considered on the entire spacecraft [17] — not just the primary piece of equipment. For example, if a mission task requires the use of the tape recorder, but the tape is full, a new task to transmit some of the data on the tape must be generated. This new task, however, cannot be modeled in the simple set up/tear down manner as is used to change configurations in the job shop environment. Instead, because this new task must use other spacecraft resources, such as the transmitter, antenna, and telecommunications processor, in the same manner as other tasks, it must be addressed in the same way as those other tasks.

3.3 Summary

Although there exist marked differences in the mission and job shop scheduling domains, the problem still devolves into one of resource allocation in a constrained environment. Automation of the scheduling process is important in both domains as a cost and efficiency measure. The major technical challenge is better control of the search space, which will lead to faster and more optimum schedules. This requirement is driven by size, complexity, and reactiveness requirements, which are common to both domains. The differences in the level of detail and type of constraints provide separate paths along which substantial beneficial information can be gathered.

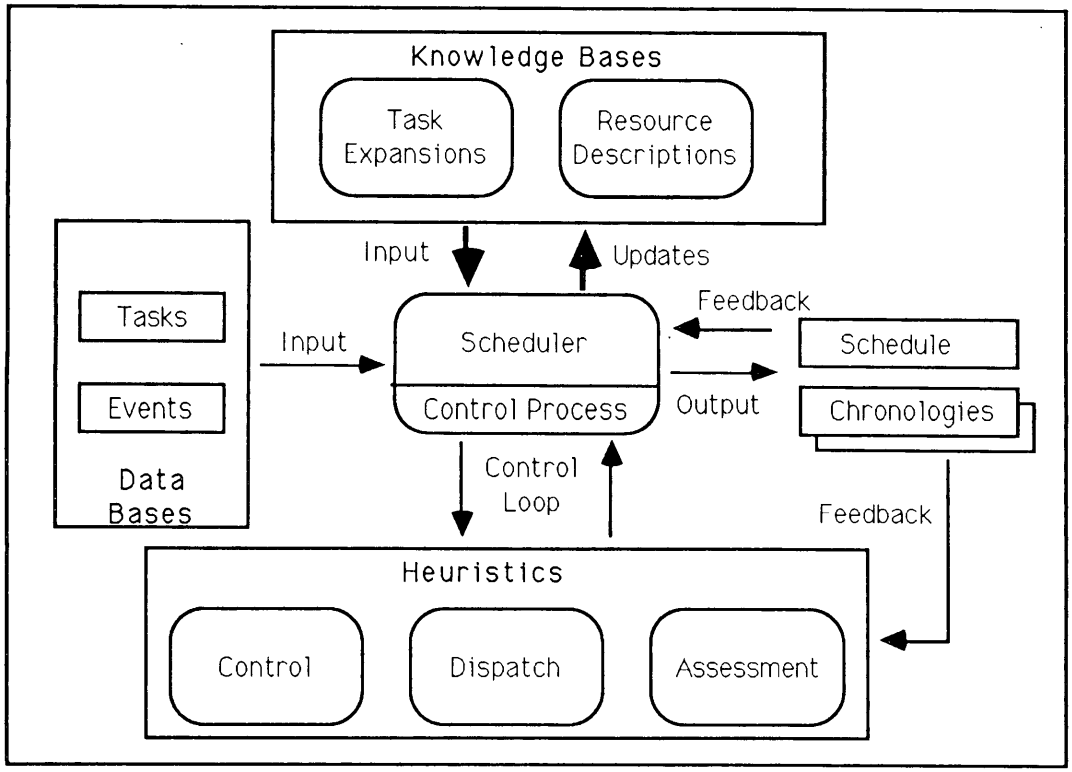
4.0 APPLICATION OF ARTIFICIAL INTELLIGENCE TECHNIQUES

Artificial Intelligence techniques are being incorporated into planning systems to help alleviate the problems associated with the size and complexity of the problem domain; the level of detail; the satisfaction of multiple, conflicting constraints; and the need to be reactive to changes in the domain environment. Special architectures and data structures are needed to support the use of these techniques. This section will present a generalized intelligent scheduling system architecture and discuss the AI techniques most applicable to automated scheduling.

4.1 Architecture to Support Automated Scheduling

One of the major benefits of the use of AI in automated planning is the decoupling of the schedule model from the scheduling engine. This allows the addition of different types of tasks and resources without requiring changes to the scheduler. A generalized view of an intelligent scheduling system is given in Figure 3. The major components of the system are the knowledge bases, the data bases, the heuristics, and the schedule itself. The information in these distinct areas are integrated by the scheduling engine which produces the actual schedule.

Figure 3. Generalized Intelligent Scheduling System



4.1.1 Knowledge and Data Bases

Knowledge bases are used to represent the static information required by the scheduler. These knowledge bases form the model of the scheduling environment. They contain information on the types of resources available and any parameters for their operation. Descriptions of the tasks, which include such parameters as length of time, appropriate equipment, and detailed task breakdowns, are also represented in the knowledge bases.

The data bases contain the actual inputs to the scheduling system, the requested tasks. Other information pertinent to the scheduler can be treated as a data base. This includes any special tasking that occurs while the schedule is executing or any special "events," such as equipment failure or environmental problems.

The information in the knowledge and data bases is maintained separately from the other components of the scheduler. This information can be updated and modified without requiring substantial changes in those other components. The primary advantage of this feature is that the scheduling system can be modified to support other, albeit similar, domains by simply switching the knowledge and data bases. In a factory environment, this would imply being able to use the same basic scheduling tool across multiple types of job shops. In the mission scheduling environment, this feature allows a single scheduler to support several different types of spacecraft.

4.1.2 Heuristics

As stated earlier, the scheduling problem devolves into controlling the search through a very large and complicated problem space. Brute-force search mechanisms are incapable of supporting automated scheduling with realistic and acceptable response times. Instead, heuristics are used to determine how to conduct the search and provide a means of beating the combinatorics problem currently affecting both mission and job shop scheduling.

Heuristics are simply rules of thumb which guide the performance of a given activity. Research at JPL has characterized three types of heuristics: (1) assessment heuristics, which assess the state of the schedule and provide information on how well the scheduler is performing; (2) dispatch heuristics, which perform the actual scheduling actions; and (3) control heuristics, which set and change the focus of attention of the scheduling process [18]; [19]. The heuristics are the "brain" of the scheduling system. They determine what areas of the schedule to concentrate on; what types of changes to make; and, based on how well the scheduler is doing, when to change approaches.

In order to control the search, the scheduler must know about the difficulties arising in the particular schedule. The scheduler must identify the problem contention areas, called bottlenecks [20]. Once this information is available, the scheduler can then use that information to direct the search process. This type of use of heuristics has been used in Ralph [21], a scheduler for the NASA Deep Space Network, and OPT and OPIS for factories.

4.2 Iterative Refinement

The common approach to scheduling is to slowly build up a schedule by adding a single task at a time to a conflict-free schedule. An alternative to this approach is the use of iterative refinement in the developing of a schedule. This technique is based on the observation of expert human schedulers which indicates that they use techniques which stress the concept of an "evolving" schedule. As they develop and refine the schedule, the human experts choose from their repertoire of scheduling techniques (heuristics) to identify and resolve problem areas in the schedule. They make multiple passes (iterative refinement) [22] over the schedule, sometimes allowing it to get worse before it improves, but continuously evolving it towards an acceptable solution. JPL efforts in scheduling are based on this model of the behavior of the expert human schedulers.

When human schedulers work on a schedule, they begin by making a rough cut of the schedule. They use this initial schedule to identify where problem areas are and characterize these problem areas. They then pick an area of the schedule to work on, which we refer to as determining the focus, and choose the most appropriate scheduling techniques from their vast repertoire.

As the scheduler works on a schedule, he or she will, at various points, focus on different aspects of the problem. The focus defines which resources, temporal regions, and tasks the scheduler needs to work on next. The focus also includes appropriate scheduling techniques which should be applied during a particular stage of the scheduling process. The scheduler monitors the effort expended on the different areas of the schedule [23]. This information is then analyzed and used to select the next focus state of the scheduling engine.

Throughout the scheduling process, the focus changes. The evolving schedule approach allows the scheduler to make simplifying assumptions. These assumptions include ignoring resources that have little impact on the final schedule and eliminating some of the possible scheduling actions. This enables the scheduler to quickly rough out a schedule which then becomes the basis for further refinement [24].

4.3 Applicability of AI Techniques

AI techniques are necessary to enable automated scheduling systems to support real-world domains such as mission scheduling. In other domains, schedulers have been willing to trade reactiveness, level of detail, or performance in their automated scheduling. The mission domain, however, is rapidly exceeding the available project resources. The cost and productivity of the human schedulers are severely limiting in light of the even larger and more complex space systems planned for the future. Yet, neither reactiveness, or level of detail, or performance can be sacrificed. The only alternative is to explore areas which could enhance the performance of automated schedulers.

Iterative refinement is the style of scheduling which is performed by expert human schedulers. The use of knowledge and data bases, and heuristic control are important aspects of the iterative refinement method. Intelligent aids which enable automated detection and

classification of scheduling problems, a suite of heuristics which provides varying levels of complexity in executing scheduling actions, and a suite of control heuristics which enables the scheduler to determine what techniques to use and when are critical to modeling expert human behavior.

5.0 CONCLUSION

The mission and job shop scheduling domains have substantial areas of overlap in terms of the problems which must be overcome in order to implement automated scheduling. Although the causes are different, both domains are severely underconstrained and require special control mechanisms to keep the search process from exploding. The level of detail required in mission scheduling is substantially higher than that in job shop scheduling, which drives the mission scheduling domain to consider more elaborate domain models. While job shop scheduling is currently being done using a combination of human and automated schedulers, it would substantially benefit from automated schedulers with the capability to handle the richer domain representation [25] critical to supporting mission scheduling.

An automated scheduler in either domain must be reactive to the environment. Because scheduling is underconstrained and input sensitive, reactive scheduling requires the old schedule as input, several scheduling techniques which can be used, and tight control on the search. AI technology is essential to providing these capabilities.

ACKNOWLEDGEMENTS

Other than the authors, individuals participating in this project include Loretta Falcone and Gaius Martin.

REFERENCES

[1] Rosenthal, D., and Jackson, R. (1988) "Development of a Pioneer Venus Expert Scheduling System" AIAA 26th Aerospace Sciences Meeting, AIAA-88-0551.

[2] Biefeld, E. (1986) "PLAN-IT: Knowledge-Based Mission Sequencing" Proceedings of SPIE on Space Station Automation pp. 126–130.

[3] Melnyk, S. "Production Control: Issues and Challenges" (1987) Proceedings from the First International Conference on Expert Systems and the Leading Edge in Production Planning and Control pp. 199–232.

[4] Dean, T. (1986a) "Intractability and Time Dependent Planning" Workshop on Planning and Reasoning About Action in Proceedings of Workshop on Planning and Reasoning About Action, p. 143–164, June 1986.

[5] Web, A. (1986) "Combinatorial Complexity of the Flight Project/Deep Space Network Resource Allocation/Scheduling Problem" Operations Research Report, Jet Propulsion Laboratory, Pasadena, California.

[6] Dean, T. (1986b) "Handling Shared Resources in a Temporal Data Base Management System" Decision Support Systems No. 2, pp. 135–143.

[7] Tate, A. (1985) "A Review of Knowledge-Based Planning Techniques" Knowledge Engineer's Review Vol. 1, No 2.

[8] Bell, C. (1985) "Resource Management in Automated Planning" AIAI-TR-8 Artificial Intelligence Applications Institute, University of Edinburgh.

[9] Minifie, R., and Davis, R. (1986) "Survey of MRP Nervousness Issues" Production and Inventory Management Vol. 27, No 2.

[10] Brown, M. (1988) "The Dynamic Rescheduler: Conquering the Changing Production Environment" Proceedings of the Fourth IEEE Conference on Artificial Intelligence Applications.

[11] Ow, P. S., Smith, S., and Thiriez, A. (1988) "Reactive Plan Revision" Proceedings of the National Conference on Artificial Intelligence, pp. 77–82.

[12] Meleton, M. P., Jr. (1986) "OPT - Fantasy or Breakthrough?" Production and Inventory Management Vol. 27, No. 2.

[13] Fox, M. and Smith, S. (1984) "ISIS: A Knowledge-Based System for Factory Scheduling" Expert Systems Vol. 1, pp. 25–49.

[14] Smith, S., Fox, M., and Ow, P. S. (1986) "Constructing and Maintaining Detailed Production Plans: Investigations into the Development of Knowledge-Based Factory Scheduling Systems: *AI Magazine*, Vol. 7, No. 4.

[15] Le Pape, C. and Smith, S. (1987) "Management of Temporal Constraints for Factory Scheduling" Proceedings of the Working Conference on Temporal Aspects in Information Systems.

[16] Vere, S. (1983) "Planning in Time: Windows and Durations for Activities and Goals" IEEE Transactions on Machine Intelligence No. 3, pp. 246–267.

[17] Miller, G., Johnston, M., Vick, S., Sponsler, J., and Lindenmayer, K. (1988) "Knowledge Based Tools for Hubble Space Telescope Planning and Scheduling: Constraints and Strategies," *Telematics and Informatics* Vol. 5, No. 3, pp. 197–212.

[18] Orelup, M., Cohen, P., Dixon, J., Simmons, M. (1988) "Dominic II: Meta-Level Control in Iterative Redesign" Proceedings of the National Conference on Artificial Intelligence pp. 22–30.

[19] Currie, K., and Tate, A. (1985) "O-Plan: Control in the Open Planning Architecture" Proceedings of the BCS Expert Systems '85 Conference.

[20] Muscettola, N. and Smith, S. (1987) "A Probabilistic Framework for Resource-Constrained Multi-Agent Planning" Proceedings of the Tenth International Joint Conference on Artificial Intelligence pp. 1063–1066.

[21] Johnson, C., and Werntz, D. (1987) "Automation of the Resource Allocation and Planning System at NASA's Jet Propulsion Laboratory" SPACE: Technology, Commerce and Communications Houston, Texas.

[22] Biefeld, E. and Cooper, L. (1988) "Replanning and Iterative Refinement in Mission Scheduling" Sixth Annual Intelligence Community AI Symposium.

[23] Biefeld, E. and Cooper, L. (1989) "Scheduling with Chronology Directed Search" To appear in the Proceedings of the AIAA Computers in Aerospace VII.

[24] Simmons, R. (1988) "A Theory of Debugging Plans and Interpretations" Proceedings of the National Conference on Artificial Intelligence, pp. 94–99.

[25] Kanet, J. (1987) "Fifth Generation Manufacturing Control" Proceedings from the First International Conference on Expert Systems and the Leading Edge in Production Planning and Control pp. 277–290.

APPENDIX J

OMP II USER INTERFACE DISPLAY PHOTOGRAPHS

The OMP User Interface Displays consist of several windows. The purposes of these windows are given below and are referenced to the diagram of the display in Figure 1. Color photographs of the actual displays follow.

1. Priority Color Key Display: Maps the priority numbers to the color coding used to represent them.

2. Current Phase: Displays the current phase in which the scheduler is operating. Possible values are Load, Resource, Bottleneck, Optimize, Alert, and Waiting.

3. Strategem: Displays the strategy which the scheduler is currently using. Possible values are: Rand(om) Alloc(ation), Shuffle, Delete, Localize, Pack, Shrink, and Undelete.

4. Messages: Displays messages from the scheduler to the user. These messages perform the following functions:

    a. Identify the tasks which the scheduler is currently deleting/undeleting.
    b. Indicate the status of the schedule at the end of a scheduling phase.
    c. Name of the files that have been loaded in.

5. Broadcast Edit: Contains the parameters of a broadcast and allows the user to edit these parameters.

6. Resource Menu: Lists the resources available and enables the user to activate the various resource displays.

7. Resource Display Window: Contains all the user-selected timeline displays.

    a. Strip Chart: Compacted version of Histogram.
    b. Histogram: Shows resource usage relative to capacity,
    c. Gantt Chart: Show the steps of the tasking assigned to a given resource.
    d. Indication Bars: Mark the bottleneck area that the scheduler is working on (blue) and identify the area of focus of the the scheduler at a given time (black).

8. Priority Status: Shows the distribution of tasks which are assigned on the schedule as a function of priority.

9. Command Interface: Allows the user to enter commands using the keyboard.

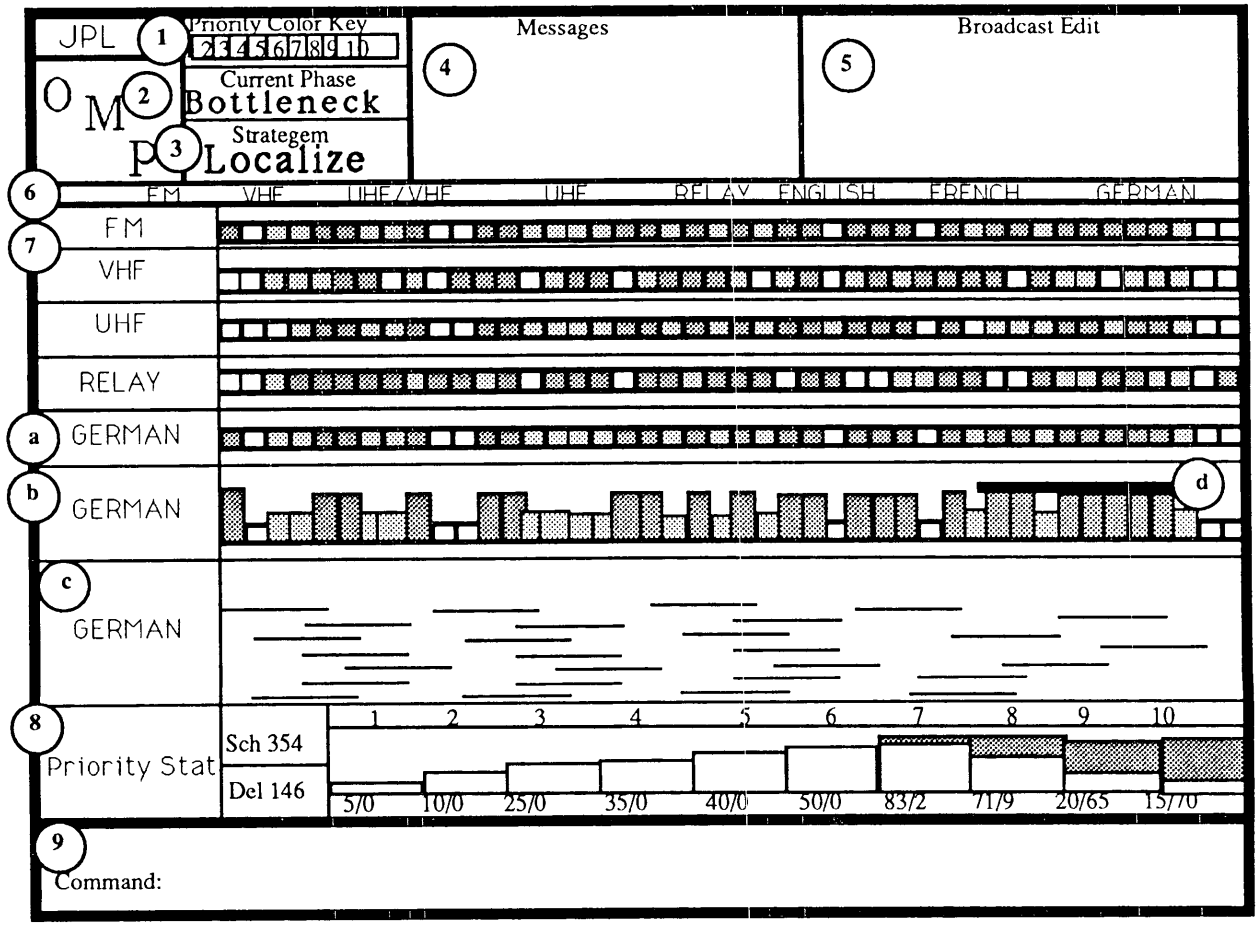
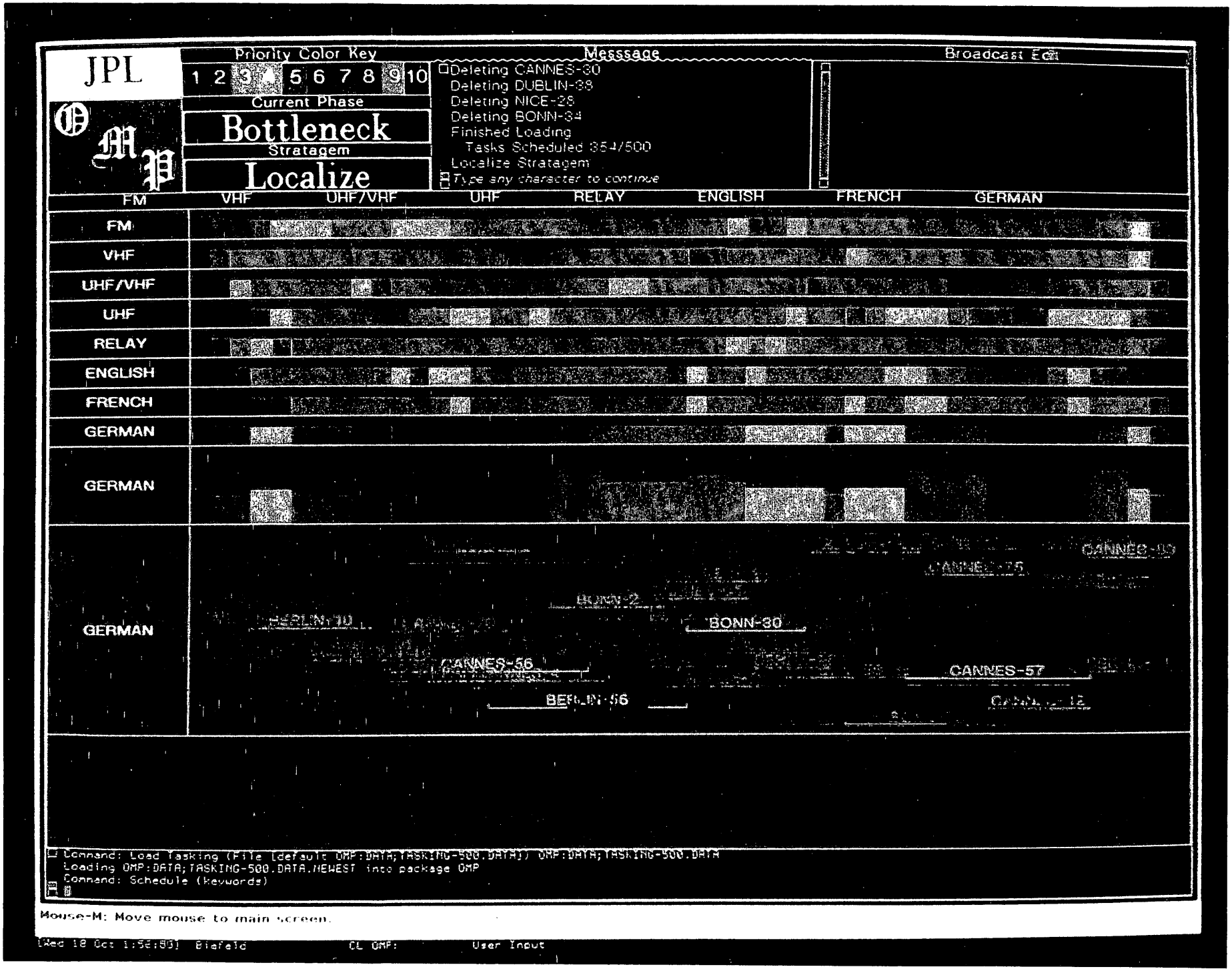
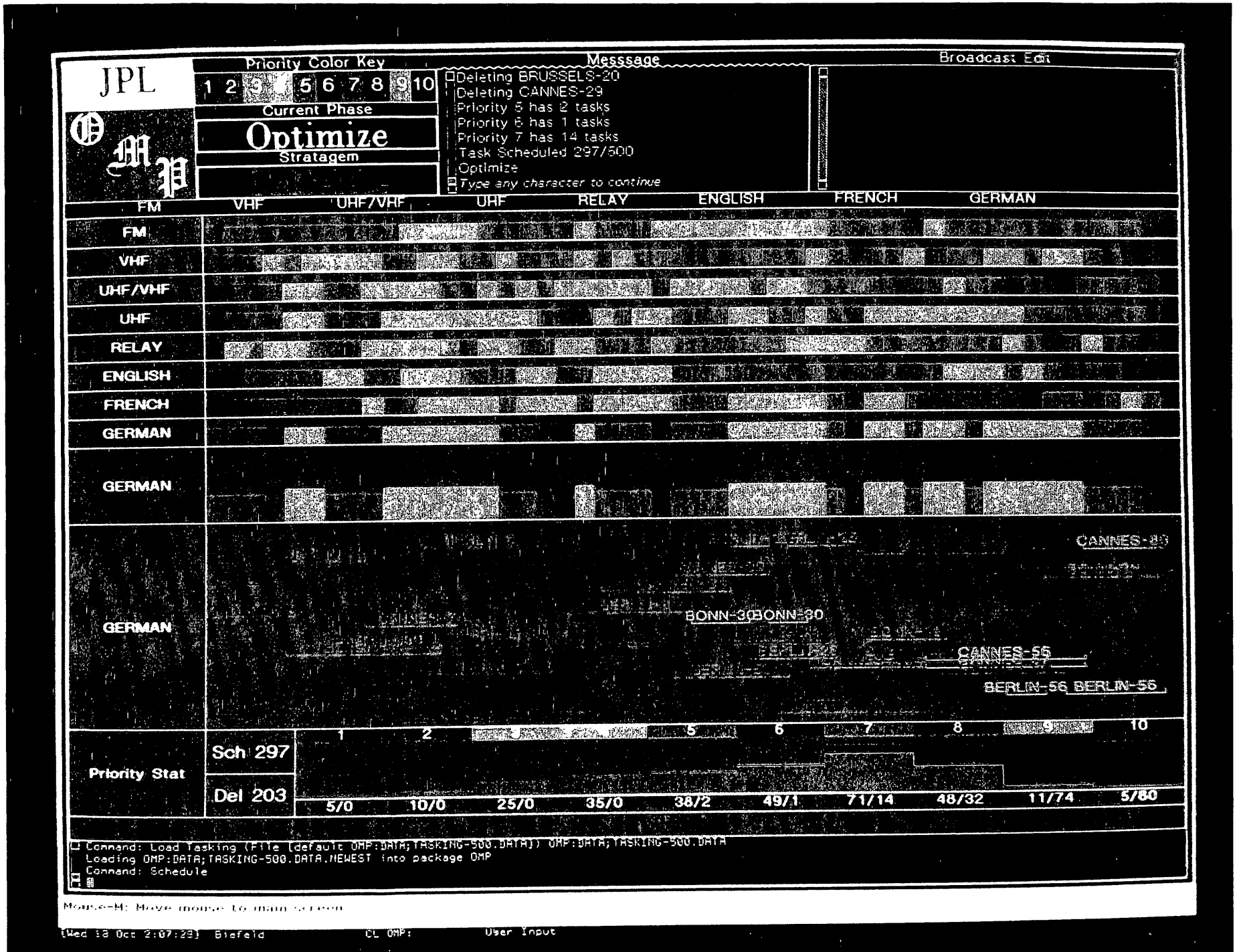
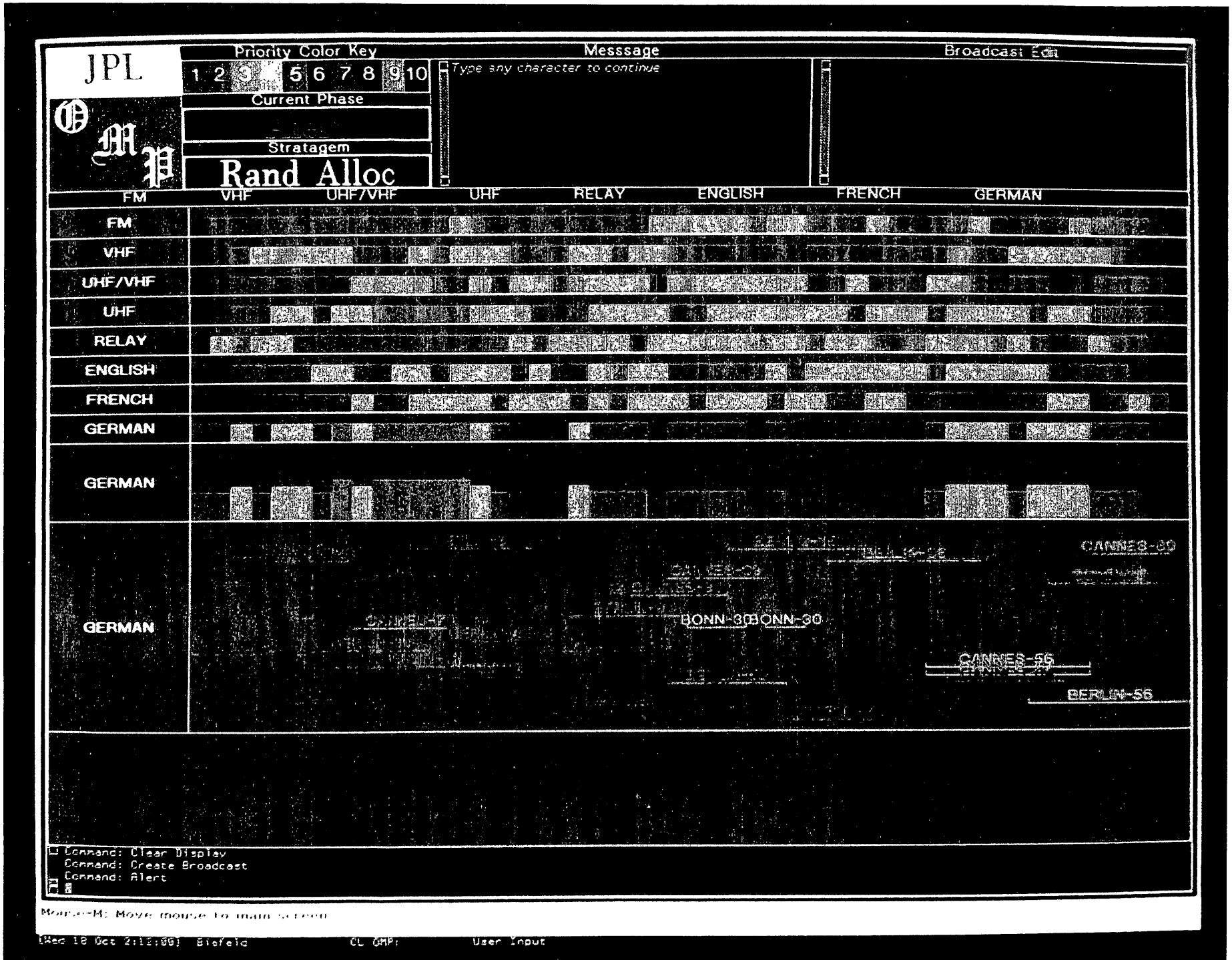


Figure 1. OMP User Interface Display







TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No. 90-16	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle OPERATIONS MISSION PLANNER: FINAL REPORT		5. Report Date March 15, 1990 6. Performing Organization Code
7. Author(s) Eric Biefeld and Lynne Cooper		8. Performing Organization Report No.
9. Performing Organization Name and Address JET PROPULSION LABORATORY California Institute of Technology 4800 Oak Grove Drive Pasadena, California 91109		10. Work Unit No. 11. Contract or Grant No. NAS7-918 13. Type of Report and Period Covered JPL Publication
12. Sponsoring Agency Name and Address NATIONAL AERONAUTICS AND SPACE ADMINISTRATION Washington, D.C. 20546		14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

The Operations Mission Planner (OMP) Final Report documents the findings of the OMP research task, which investigated the applicability of Artificial Intelligence (AI) technology in support of automated scheduling. This report summarizes the goals of the effort and highlights the technical accomplishments. The OMP task succeeded in identifying how AI technology could be applied and demonstrated an AI-based automated scheduling approach through the OMP prototypes.

17. Key Words (Selected by Author(s))	18. Distribution Statement
Ground Support Systems and Facilities (Space) Computer Programming and Software	Unclassified, Unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	171	